2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

NanoFlowNet: Real-time Dense Optical Flow on a Nano Quadcopter

Rik J. Bouwmeester, Federico Paredes-Vallés and Guido C. H. E. de Croon

***Abstract*— Nano quadcopters are small, agile, and cheap platforms that are well suited for deployment in narrow, cluttered environments. Due to their limited payload, these vehicles are highly constrained in processing power, rendering conventional vision-based methods for safe and autonomous navigation incompatible. Recent machine learning developments promise high-performance perception at low latency, while dedicated edge computing hardware has the potential to augment the processing capabilities of these limited devices. In this work, we present NanoFlowNet, a lightweight convolutional neural network for real-time dense optical flow estimation on edge computing hardware. We draw inspiration from recent advances in semantic segmentation for the design of this network. Additionally, we guide the learning of optical flow using motion boundary ground truth data, which improves performance with no impact on latency. Validation results on the MPI-Sintel dataset show the high performance of the proposed network given its constrained architecture. Additionally, we successfully demonstrate the capabilities of NanoFlowNet by deploying it on the ultra-low power GAP8 microprocessor and by applying it to vision-based obstacle avoidance on board a Bitcraze Crazyflie, a 34 g nano quadcopter.**

I. INTRODUCTION

Safe and reliable navigation of autonomous aerial systems in narrow, cluttered, GPS-denied, and unknown environments is one of the main open challenges in the field of robotics. Because of their small size and agility, micro air vehicles (MAVs) are optimal for this task [1], [2]. Nano quadcopters are a variety of MAVs that are characterized by minimal weight (approx. 30 g) and size (approx. 10 cm rotor-to-rotor) and hence are well suited for deployment under the aforementioned conditions. With the right algorithm design, these nano quadcopters have been demonstrated to be able to perform complex tasks such as exploration [3] or gas source seeking [4]. However, conventional approaches to these problems rely on computationally expensive "map-based" methods that require an array of sensors (e.g., stereo camera, LiDAR) and processors that, in the majority of cases, exceed the payload capacity of these vehicles.

The main approach to autonomous flight of MAVs is based on monocular vision, since a single camera can be lightweight and energy-efficient, while providing rich information on the environment. One of the most important monocular visual cues for navigation is optical flow. Until now, it has been extensively exploited on aerial vehicles with relatively high payload capacity for tasks such as obstacle avoidance [5], [6], and several bio-inspired methods for autonomous navigation [7]–[11].

All authors are with the Micro Air Vehicle Laboratory, Faculty of Aerospace Engineering, Delft University of Technology, Delft, The Netherlands. Contact: G.C.H.E.deCroon@tudelft.nl

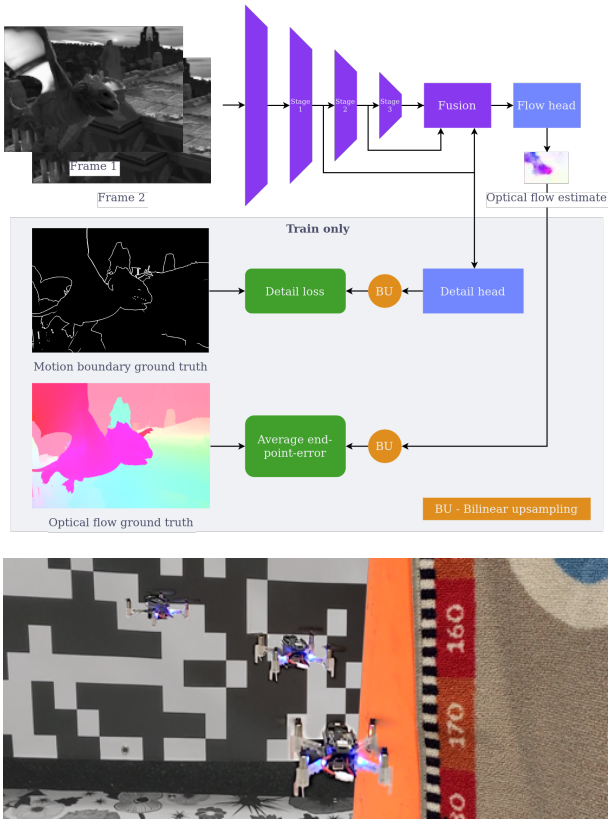


Fig. 1: *Top*: NanoFlowNet consists of (i) an encoder that extracts features from the input images, (ii) a fusion module that combines features from different levels, and (iii) a motion-boundary-guided detail head, which is only enabled during training, to guide the learning with zero cost to inference latency. *Bottom*: We demonstrate NanoFlowNet in an obstacle avoidance application on board a nano quadcopter (time-lapse image).

Traditionally, the task of monocular optical flow estimation has been performed by hand-crafted methods [12], [13]. However, the field recently shifted toward deep learning approaches [14]–[25], which deliver not only a better performance than the conventional methods but also a faster runtime. Although the focus has largely been on improving performance, efforts have been made to find models of reduced size and faster inference [15], [16], [18], [19], [22], [23], [26]. However, these methods remain computationally expensive, with runtime ranging from several to tens of frames per second (FPS) on desktop GPUs and requiring millions of parameters (and hence large amounts of memory), rendering these models incompatible with edge hardware.

In this work, instead of improving the accuracy of state-of-the-art approaches, we focus on their inference speed and, more particularly, on the deployment of a dense optical

979-8-3503-2365-8/23/$31.00 ©2023 IEEE

flow network on edge devices. To this end, we present *NanoFlowNet*, a lightweight convolutional neural network (CNN) architecture for optical flow estimation that, inspired by the semantic segmentation network STDC-Seg [27], achieves real-time inference on the ultra-low power GAP8 multi-core microprocessor on the Bitcraze AI-deck. An overview of the proposed network architecture and its training pipeline can be found in Fig. 1.

The key contributions of this paper are listed as follows. First, we introduce NanoFlowNet, a novel lightweight neural network architecture that performs, for the first time, real-time dense optical flow estimation on edge hardware. We validate this network, which runs at 5.5-9.3 FPS on the tiny GAP8 microprocessor, through extensive quantitative and qualitative evaluations on multiple datasets. Second, we show, for the first time, that using motion boundary ground truth to guide the learning of optical flow improves performance while having zero impact on inference latency. Last, we demonstrate the proposed NanoFlowNet in a real-world obstacle avoidance application on board a Bitcraze Crazyflie nano quadcopter.

The remainder of this paper is organized as follows. Section II provides an overview of the state-of-the-art on autonomous navigation of nano quadcopters and on real-time inference with CNNs. In Section III, we present the details of the proposed architecture and its training pipeline, while Section IV covers the setup of the experiments as well as the obtained results. Finally, concluding remarks are given in Section V alongside recommendations for future work.

II. Related Work

A. Autonomous navigation of nano quadcopters

The limited computational capacity of nano quadcopters (and MAVs in general) puts a constraint on the types of methods that can be used for autonomous navigation. Methods demonstrated on board nano quadcopters can be broadly grouped in model-based reinforcement learning for hovering [28], obstacle avoidance based on dedicated laser ranging sensors [3], [4], [29], and self-motion estimation using optical flow from dedicated optical flow sensors [30] or estimated with external, multi-camera setups [31], [32]. Other methods circumvent the computational constraints of these vehicles by running methods off-board [33]–[35].

Regarding edge computing hardware, recent works have focused on augmenting the computational power of nano quadcopters without exceeding their payload limitations. Methods based on application-specific integrated circuits (ASICs) [36]–[39] can efficiently provide information for specific tasks such as SLAM and visual-inertial odometry but have not yet been presented on a flying drone. More recently, parallel ultra-low power processors introduce energy-efficient multi-core processing to parallelize visual workloads on edge devices [40]. In this work, we exploit the commercially available off-the-shelf AI-deck from Bitcraze, equipped with the GreenWaves GAP8 system-on-chip (SoC) and an ultra-low power grayscale camera. This nine-core SoC has been used for several end-to-end methods that integrate perception and navigation by directly regressing inputs through a CNN into control commands [40]–[42]. Instead, in our approach, we calculate optical flow as an intermediate step. This gives us direct control over vehicle behavior and can support multiple optical-flow-based tasks to be performed simultaneously or interchangeably. Our work, motivated by these benefits, is the first to present a fully convolutional neural network for a dense (i.e., per-pixel) prediction task on board the AI-deck.

B. Real-time dense inference with CNNs

For the design of NanoFlowNet, we draw inspiration from recent semantic segmentation literature in order to significantly speed up optical flow estimations while retaining performance. More specifically, we draw inspiration from the BiSeNet [43] and STDC-Seg [27] architectures. First, BiSeNet identified a sacrifice of low-level spatial information in previous real-time methods and improved performance by proposing a multi-path architecture in which low-level spatial information is encoded in a separate path. A feature fusion module was proposed to fuse information from the high- and low-level paths, while an attention refinement module refined features through channel attention. Then, the STDC-Seg architecture introduced the STDC module, which increases the receptive field size per layer at a low computational cost. Furthermore, it identified that BiSeNet's spatial path pronounces edges, and replaced the convolutions from the path with a train-time-only "detail head" and "detail loss" to mimic the information passed from the removed convolutions, thus shrinking the model and decreasing latency. The "detail guidance ground truth" was generated by convolving the ground truth segmentation map with a Laplacian kernel.

A few elements of STDC-Seg and BiSeNet have been separately investigated in the context of optical flow. AD-Net [44] showed that channel attention can be beneficial for optical flow estimation, while EDOF [45] fused features from an edge-detector network and an optical flow encoder network for detail-guided optical flow estimation. Similar to STDC-Seg, we use edges to guide the learning.

III. Method

For the design of NanoFlowNet, we adopt the STDC-Seg network [27] and modify it to our needs. We replace all regular convolutions with depthwise separable convolutions, and we globally reduce the number of filters by a factor of four to further reduce latency and the number of parameters. We introduce an even smaller model with half of NanoFlowNet's filters (globally) and call it *NanoFlowNet-s*. Further modifications to the architecture are discussed in detail in the following subsections.

A. Motion boundary detail guidance

The closest analogy to detail guidance as used in STDC-Seg is to generate edges from the optical flow ground truth. Instead, we replace this "edge-detect" detail guidance ground truth with motion boundary ground truth from the optical flow datasets. We adopt the Focal Loss [46] to counter the class imbalance problem.

B. Strided STDC module redesign

We modify the strided STDC modules from STDC-Seg [27] to further decrease latency. The original and modified strided STDC modules can be found in Figs. 2 and 3, respectively. First, following the insights of several low-latency literature methods [47]–[51], we replace all convolutions in the STDC module with depthwise separable convolutions due to their low computational expense. Second, we identify that the first operation in the strided STDC module (a pointwise convolution) is the most expensive in terms of the number of multiply-accumulate (MAC) operations. By relocating this operation to the bottom path after the average pooling operation, we make the strided STDC block computationally more tractable overall while increasing the number of features in the top path and the number of features with a large receptive field size in the concatenated output. Our modified blocks lead to a reduction of over 50% of the MAC operations in stage 1, and of over 10% in stages 2 and 3.

C. Reduced input/output dimensionality

We design the network for low-resolution input and downscale all dataset's input frames, optical flow, and motion boundary ground truth accordingly. The scaling factor is picked such that the resulting data resolution closely matches the target application resolution (approx. qqVGA, 160x120 pixels). Horizontal and vertical scalings are identical, to fix the aspect ratio in an attempt to retain naturalism. This allows us to make the network shallower by dropping the first (expensive) convolution altogether and thus decrease latency while maintaining feature sizes in the deepest layers. The downscaled training data matches the low-resolution cameras found on nano quadcopters more closely, making our synthetic dataset more naturalistic for our intended application. As an added benefit, working with downscaled data significantly speeds up training. The primary downside of reduced input resolution is the loss of information, in particular we will miss out on small objects and small displacements that are not captured by the resolution. To be able to compare with existing optical flow works, we benchmark performance at native dataset resolution, since downscaling of flow magnitudes results in lower endpoint error (EPE) without a qualitative improvement.

Lastly, we design our network for grayscale input images, saving two third of the on-board memory dedicated to the input frames and decreasing the computational cost of the first layer (at a loss of color information).

IV. Experiments

A. Implementation details

All models are trained for 300 epochs on FlyingChairs2 [14], [52], a regenerated FlyingChairs dataset with motion boundary ground truth. We use the Adam optimizer [53], with learning rate 1e-3 and a batch size of 8. After this, we fine-tune our architectures on FlyingThings3D [54] for 200 epochs with a learning rate of 1e-4.

Given the scaling and conversion to grayscale of the input data, our network is not directly comparable with results reported by other works. For comparison, we retrain one of the fastest networks in literature, Flownet2-s [16], on the same data. Given the reduction in resolution, we drop the deepest two layers to maintain a reasonable feature size in the deepest layers, and name the model *Flownet2-xs*.

We run all experiments in a docker environment with TensorFlow 2.8.0, CUDA 11.2, CUDNN 8.1.0, TensorRT 7.2.2 on an NVIDIA GeForce GTX 1070 Max-Q with batch size 1 for benchmarking latency.

B. Performance and latency on public benchmarks

We evaluate the trained networks on the unseen MPI Sintel train subset, on both the clean and final pass. Quantitative results can be found in Table I. Regarding accuracy, according to these results, our NanoFlowNet performs better than the squeezed FlowNet2-xs architecture, despite using less than 10% of the parameters. With respect to runtime, FlowNet2-xs

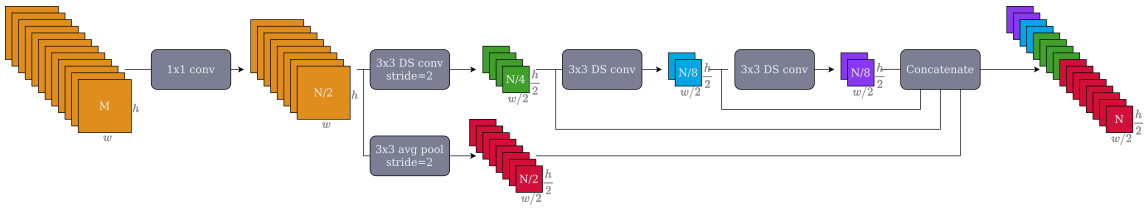


Fig. 2: Original strided STDC module from [27], with the exception that we use depthwise separable (DS) convolutions in place of all non-pointwise convolutions. We use ReLU activations after all layers in the block. M denotes the number of input features, while N is the number of output features.

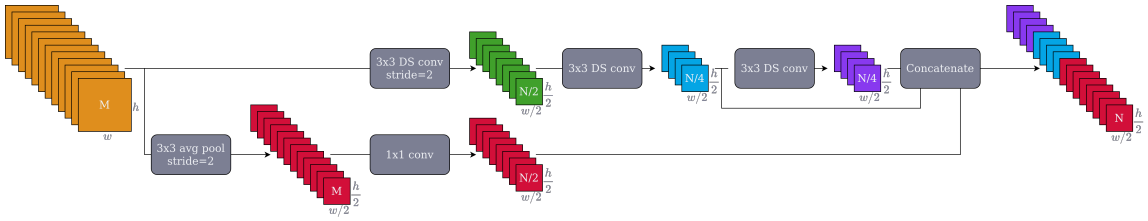
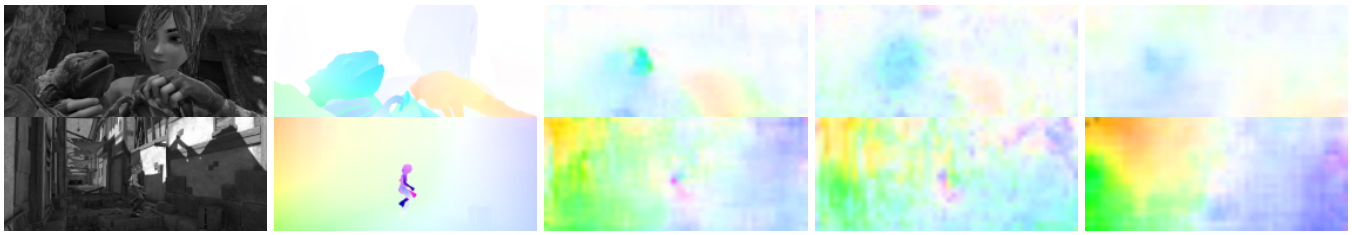


Fig. 3: Our modified strided STDC module. We reorganize the operations to minimize the spatial resolution pointwise convolutions have to perform on.



(a) Input frame I (b) Ground truth (c) NanoFlowNet (d) NanoFlowNet-s (e) FlowNet2-xs

Fig. 4: Qualitative comparison of optical flow estimates by NanoFlowNet(-s) and FlowNet2-xs on MPI Sintel (train) clean pass.

Method	**MPI Sintel (train) [EPE]**		**Frame rate [FPS]**		**Parameters**
	Clean	***Final***	***GPU*[1]**	***GAP8*[2]**	
FlowNet2-xs	9.054	9.458	150	-	1,978,250
NanoFlowNet (ours)	**7.122**	**7.979**	141	5.57	170,881
NanoFlowNet-s (ours)	9.559	10.047	**151**	**9.34**	46,749

TABLE I: Quantitative results on MPI Sintel. [1]At a resolution of 96x224. [2]At a resolution of 112x160, including vision thread.

Detail guidance method	**MPI Sintel (train)**	
	Clean	***Final***
None	7.636	8.119
Edge detect	7.404	8.141
Motion boundaries	**7.122**	**7.979**

TABLE II: Quantitative comparison of different methods of detail guidance.

Strided STDC block	**MPI Sintel (train) [EPE]**		**Frame rate [FPS]**	
	Clean	***Final***	***GPU*[1]**	***GAP8*[2]**
Unmodified	7.483	8.114	136	4.84
Modified	**7.122**	**7.979**	**141**	**5.57**

TABLE III: Quantitative comparison of the original and the modified strided STDC block. [1]At a resolution of 96x224. [2]At a resolution of 112x160, including vision thread.

Mode	**MPI Sintel (train) [EPE]**		Frame rate [FPS]	
	Clean	***Final***	***GPU*[1]**	***GAP8*[2]**
Color	7.726	8.344	**141**	5.18
Grayscale	**7.122**	**7.979**	**141**	**5.57**

TABLE IV: Quantitative comparison of grayscale vs. color input frame-based architectures. [1]At a resolution of 96x224. [2]At a resolution of 112x160, including vision thread.

does not fit on the GAP8 microprocessor due to the network size (i.e., lack of memory). To put the achieved latency of NanoFlowNet in perspective, we execute FlowNet2-xs' first two convolutions and the final prediction layer on the GAP8. The three-layer architecture achieves 4.96 FPS, which is slower than running the entire NanoFlowNet (5.57 FPS). On laptop GPU hardware, NanoFlowNet achieves comparable FPS to FlowNet2-xs. NanoFlowNet-s has lower performance than both other models, but has a low parameter count with only 27% of NanoFlowNet's and 2.4% of FlowNet2-xs's parameters, and is the fastest out of all the networks tested.

Qualitative results, presented in Fig. 4, confirm that NanoFlowNet makes the most accurate optical flow estimates out of all the networks tested. Interestingly, both NanoFlowNet and NanoFlowNet-s appear to detect displacements of smaller objects, which FlowNet2-xs misses. However, NanoFlowNet-s' flow estimates are highly noisy.

C. Ablation study

1) Motion boundaries detail guidance: We verify the effectiveness of motion boundary detail guidance by retraining two additional networks, one with detail guidance based on the optical flow ground truth convolved with a Laplacian kernel (further referred to as "edge-detect guidance"), and another one with no detail guidance. Quantitative and qualitative results can be found in Table II and Fig. 5, respectively. As shown, motion boundary detail guidance improves results and outperforms edge detect detail guidance. Since all these guidance methods only affect (i.e., guide) the training behavior, all methods have identical latency. Qualitative results show that motion-boundary-guided optical flow best defines moving objects, and shows the least "leakage" of foreground objects into the background.

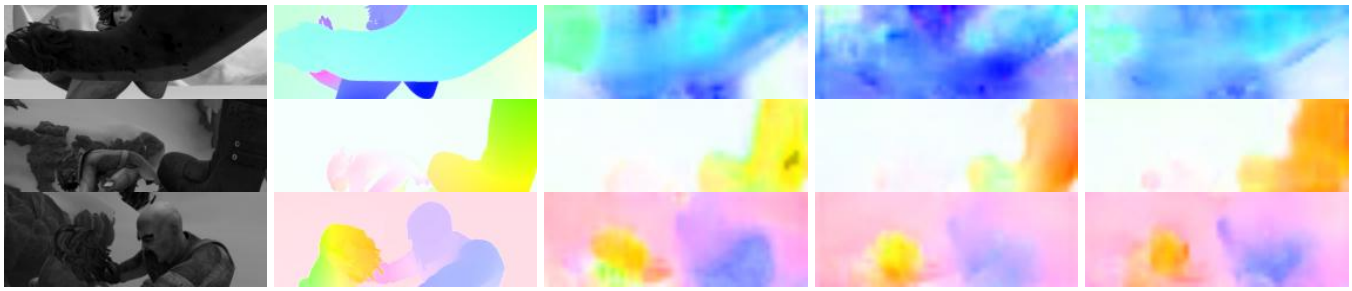
2) Strided STDC module redesign: Table III shows the effects of the strided STDC module redesign. The network with the redesigned module is both faster (both on laptop GPU and the GAP8 microprocessor) and more accurate.

3) Reduced input dimensionality: A comparison between training and inferring on grayscale images compared to color images can be found in Table IV. Our grayscale model outperforms the color variant. We hypothesize that this is due to the limited capacity of the network. The latency of the grayscale model on the GAP8 is lower due to reduced data transfer and a cheaper first convolution.

D. Obstacle avoidance implementation

We deploy the proposed NanoFlowNet architecture on a Crazyflie 2.x equipped with the AI-deck and the flow-deck for the task of vision-based obstacle avoidance. We use the AI-deck to capture images with the front-facing camera and to run optical flow inference and processing. The downward-facing optical flow deck is used for positioning only. The total flight platform weighs in at 34 g. See Fig. 6 for a picture of the platform.

1) Control strategy: We implement the horizontal balance strategy from [55], [56], with which the yaw rate $\dot{\psi}$ is set based on the error e_{rl} between the sum of flow magnitudes in the left and right half of the flow estimate (see Eq. 1). We



(a) Input frame I (b) Ground truth (c) Motion-boundary guidance (d) Edge-detect guidance (e) No detail guidance

Fig. 5: Qualitative comparison of different detail guidance methods on MPI Sintel (train) clean pass.

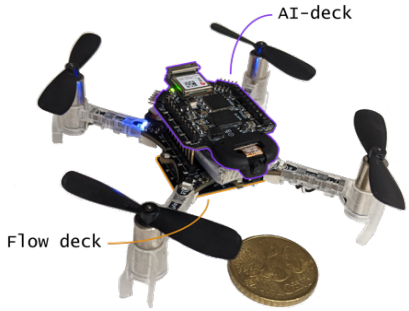


Fig. 6: Crazyflie 2.x equipped with (i) the AI-deck used for image acquisition using a front-facing camera and to run optical flow inference, and (ii) the downward-facing flow-deck used only for positioning.

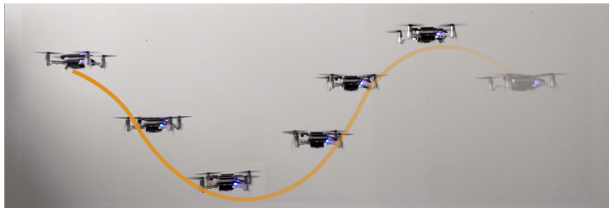


Fig. 7: Inspired by GapFlyt [6], we deliberately let the quadcopter oscillate vertically to generate additional optical flow.

set gains $k_p = 0.0126$ and $k_d = 0.0018$ experimentally. The forward velocity of the quadcopter is set at 0.2 m/s.

$$\dot{\psi} = k_p e_{rl} + k_d \dot{e_{rl}} \tag{1}$$

We augment the balance strategy by implementing active oscillations (a cyclic up-down movement, see Fig. 7) which results in additional optical flow being generated across the field of view (FOV). This is particularly helpful for avoiding objects in the direction of horizontal travel. Up-down rather than left-right surveying favors detecting obstacles wider than taller in nature, but is much simpler to combine with the left-right balance strategy. Additionally, left-right surveying requires rolling, which introduces rotational flow that does not contain depth information.

We implement both the CNN and calculation of e_{rl} on the GAP8 microprocessor of the AI-deck. Calculating the flow error on the AI-deck significantly reduces the amount of data that needs to be transmitted over UART to the autopilot. The calculation of the yaw rate is done on the Crazyflie 2.x, and fed into the controller.

2) AI-deck implementation: The CNN processing power on the AI-deck comes from the GreenWaves Technologies GAP8. The chip is organized around the central single-core fabric controller (FC) and the eight-core cluster (CL) for parallelized workloads. For our application, we run FC@250MHz, CL@230MHz, and VDD@1.2V.

Our AI-deck is equipped with the HM01B0 monochrome camera, which supports a resolution of up to 324x324, a QVGA (244x324) window mode, a 2x2 monochrome binning mode, and cropping. For our application we enable both the window mode and binning mode (122x162) and take a central crop of 112x160, to ensure a matched spatial resolution of upsampled and skipped features in the network architecture. At our input resolution, using grayscale versus color reduces the L2 memory usage on the AI-deck by 14%. This additional L2 memory is made available to the AutoTiler, which improves inference time by reducing the number of data transfers.

In this work, we utilize the GreenWaves Technologies GAPflow toolset for porting our CNN to the GAP8. NNTool takes a TensorFlow Lite or ONNX CNN description and maps all operations and parameters to a representation compatible with AutoTiler, the GAPflow tiling solver.

We use NNTool to implement 8-bit post-training quantization to our CNN. We quantize on images from the MPI Sintel dataset [57] and achieve an average signal to quantization noise ratio (SQNR) of 10.

3) Experimental setup: We compose two indoor environments for obstacle avoidance. First, an open environment, with obstacles exclusively placed at the outline of the environment. Second, a cluttered environment, with obstacles placed throughout (see Fig. 9). Obstacles include textured and untextured poles, synthetic plants, flags, or panels. Both environments are enclosed with textured panels to trap the quadcopter inside. Panel textures consist of forest texture, data matrix texture, and a drone racing gate texture. In both environments, we augment the enclosure's texture with highly textured mats and curtains.

The simple proof-of-concept control algorithm has no dedicated method of dealing with head-on collisions. By placing obstacles around the perimeter of the open environment we minimize the risk of a head-on collision with the panels as they introduce an imbalance of optical flow, even on a fully perpendicular collision path with a panel.

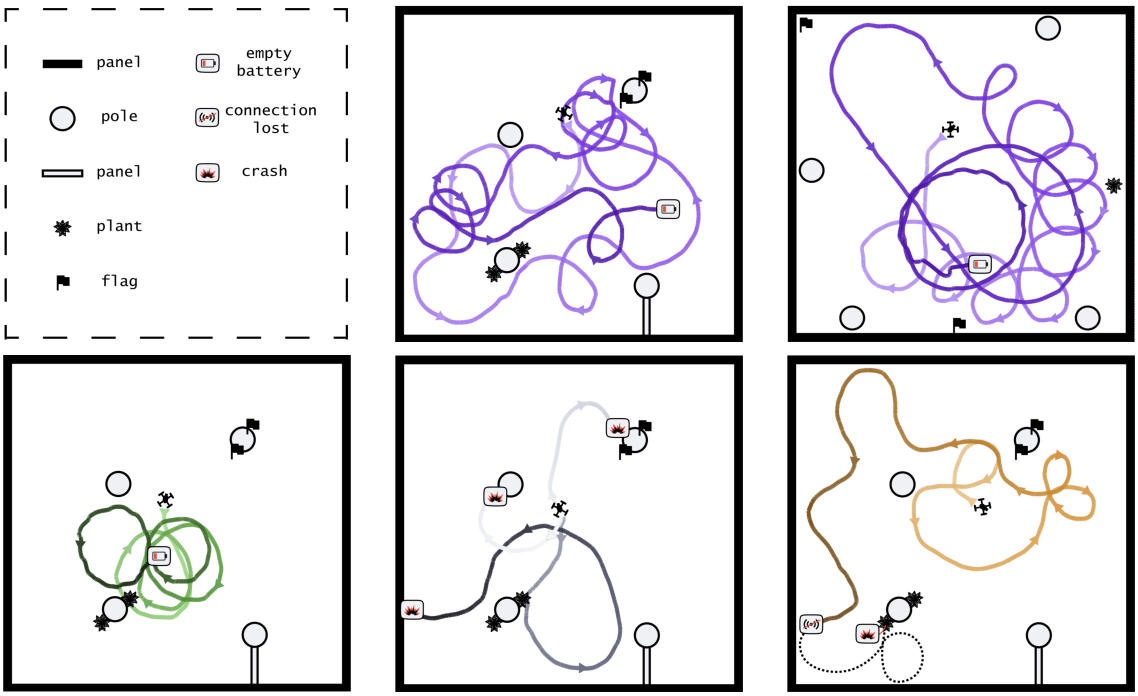


Fig. 8: Results of multiple obstacle avoidance runs in cluttered and open environments. Position recorded with an OptiTrack Motion Capture System.

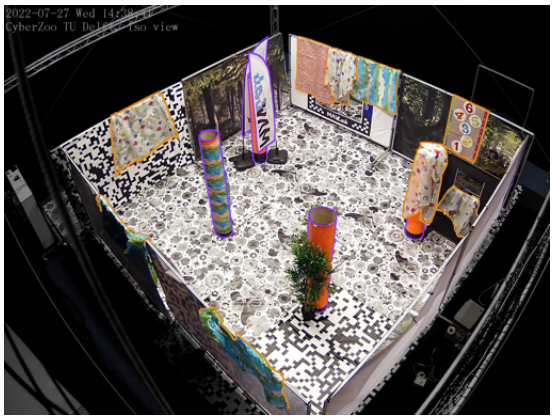


Fig. 9: Overview of the cluttered, obstacle avoidance environment. Obstacles are outlined in purple, while texture-enhancing mats and curtains in orange.

For each experiment, we start the quadcopter at approximately the same location, with varying heading. We let the quadcopter run until a collision or empty battery. We record flight positioning data with an OptiTrack Motion Capture System for post-flight analysis only and record experiments with an ISO view and top view camera.

4) Results: Flight paths extracted from the motion capture system are plotted on maps of the environment and can be found in Fig. 8. The control algorithm is most robust in the open environment, with the quadcopter managing to drain a full battery without crashing. In the cluttered environment, performance is much more variable. Especially in occasions where obstacles are in close proximity to one another, the quadcopter tends to successfully avoid an obstacle, only to collide with another during the maneuver. Adding a head-on collision detection based on the detection of the focus-of-expansion (FOE) and divergence estimation (e.g., [11]) could help avoid obstacles in these cases.

In several of the successful avoidances, the quadcopter initially responds weakly to the obstacle, only to turn away more harshly when the course has already been corrected sufficiently. This behavior is expected because of two reasons. First, the optical flow due to forward movement is zero at the FOE and maximum at the edge of the peripheral vision. Second, due to the fact that the obstacles take up more of the FOV when they are in closer proximity to the quadcopter, they generate more optical flow. This behavior could be corrected by weighing the optical flow more heavily towards the center of the image.

Another notable feature of the flight paths is that the nano quadcopter frequently appears to enter a spiraling path. The control algorithm is overreacting to stimuli from across the environment. Despite this, the behavior is consistent, the resulting paths are still exploring the environments, and the nano quadcopter is able to break out of the spiraling motion by approaching a panel (see Fig. 8, top right) or approaching an obstacle (see Fig. 8, top center).

V. Conclusions & Discussion

In this work, we introduced a lightweight CNN architecture for dense optical flow estimation on edge hardware, called NanoFlowNet. We achieved real-time latency on the AI-deck. Furthermore, we showed that training our network guided on motion boundaries improves performance at zero cost to latency. Finally, we implemented NanoFlowNet in a real-world obstacle avoidance application on board a Bitcraze Crazyflie nano quadcopter. For future work, we expect examples that take more advantage of the dense information in the generated optical flow field.

REFERENCES

[1] D. Floreano and R. J. Wood, "Science, technology and the future of small autonomous drones," *Nature*, vol. 521, no. 7553, pp. 460–466, may 2015.

[2] B. Bodin, H. Wagstaff, S. Saecdi, L. Nardi, E. Vespa, J. Mawer, A. Nisbet, M. Lujan, S. Furber, A. J. Davison, P. H. Kelly, and M. F. O'Boyle, "SLAMBench2: Multi-Objective Head-to-Head Benchmarking for Visual SLAM," in *Proceedings - IEEE International Conference on Robotics and Automation*, sep 2018, pp. 3637–3644.

[3] K. N. McGuire, C. de Wagter, K. Tuyls, H. J. Kappen, and G. C. H. E. de Croon, "Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment," *Science Robotics*, vol. 4, no. 35, oct 2019.

[4] B. P. Duisterhof, S. Li, J. Burgues, V. J. Reddi, and G. C. H. E. de Croon, "Sniffy Bug: A Fully Autonomous Swarm of Gas-Seeking Nano Quadcopters in Cluttered Environments," in *IEEE International Conference on Intelligent Robots and Systems*, jul 2021, pp. 9099–9106.

[5] P. Gao, D. Zhang, Q. Fang, and S. Jin, "Obstacle avoidance for micro quadrotor based on optical flow," in *Proceedings of the 29th Chinese Control and Decision Conference, CCDC 2017*, jul 2017, pp. 4033–4037.

[6] N. J. Sanket, C. D. Singh, K. Ganguly, C. Fermuller, and Y. Aloimonos, "GapFlyt: Active vision based minimalist structure-less gap detection for quadrotor flight," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2799–2806, 2018.

[7] J. Conroy, G. Gremillion, B. Ranganathan, and J. S. Humbert, "Implementation of wide-field integration of optic flow for autonomous quadrotor navigation," in *Autonomous Robots*, vol. 27, no. 3, oct 2009, pp. 189–198.

[8] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "MAV navigation through indoor corridors using optical flow," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 3361–3368.

[9] G. C. H. E. de Croon, "Monocular distance estimation with optical flow maneuvers and efference copies: A stability-based strategy," *Bioinspiration and Biomimetics*, vol. 11, no. 1, jan 2016.

[10] J. R. Serres and F. Ruffier, "Optic flow-based collision-free strategies: From insects to robots," *Arthropod Structure and Development*, vol. 46, no. 5, pp. 703–717, sep 2017.

[11] G. C. H. E. de Croon, C. De Wagter, and T. Seidl, "Enhancing optical-flow-based control by learning visual appearance cues for flying robots," *Nature Machine Intelligence*, vol. 3, no. 1, pp. 33–41, jan 2021.

[12] B. D. Lucas and T. Kanade, "Iterative Image Registration Technique With an Application To Stereo Vision." in *Proc 7th Intl Joint Conf on Artificial Intelligence*, 1981, pp. 674–679.

[13] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185–203, aug 1981.

[14] A. Dosovitskiy, P. Fischery, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. V. D. Smagt, D. Cremers, and T. Brox, "FlowNet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2758–2766.

[15] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, nov 2017, pp. 2720–2729.

[16] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1647–1655.

[17] S. Zhao, X. Li, and O. El Farouk Bourahla, "Deep optical flow estimation via multi-scale correspondence structure learning," in *IJCAI International Joint Conference on Artificial Intelligence*, vol. 0, jul 2017, pp. 3490–3496.

[18] D. Sun, X. Yang, M. Y. Liu, and J. Kautz, "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2017, pp. 8934–8943.

[19] T. W. Hui, X. Tang, and C. C. Loy, "LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8981–8989.

[20] Z. Yin, T. Darrell, and F. Yu, "Hierarchical discrete distribution decomposition for match density estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, dec 2019, pp. 6037–6046.

[21] G. Yang and D. Ramanan, "Volumetric correspondence networks for optical flow," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[22] T. W. Hui, X. Tang, and C. C. Loy, "A Lightweight Optical Flow CNN - Revisiting Data Fidelity and Regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2555–2569, feb 2021.

[23] T. W. Hui and C. C. Loy, "LiteFlowNet3: Resolving Correspondence Ambiguity for More Accurate Optical Flow Estimation," in *European Conference on Computer Vision*, 2020, pp. 169–184.

[24] S. Zhao, Y. Sheng, Y. Dong, E. I. Chang, and Y. Xu, "Maskflownet: Asymmetric feature matching with learnable occlusion mask," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, mar 2020, pp. 6277–6286.

[25] Z. Teed and J. Deng, "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow (Extended Abstract)," in *European Conference on Computer Vision*, aug 2020, pp. 402–419.

[26] J. Hur and S. Roth, "Iterative residual refinement for joint optical flow and occlusion estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, apr 2019, pp. 5747–5756.

[27] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking BiSeNet For Real-time Semantic Segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, apr 2021, pp. 9711–9720.

[28] N. O. Lambert, D. S. Drew, J. Yaconelli, S. Levine, R. Calandra, and K. S. Pister, "Low-Level Control of a Quadrotor with Deep Model-Based Reinforcement Learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4224–4230, jan 2019.

[29] B. P. Duisterhof, S. Krishnan, J. J. Cruz, C. R. Banbury, W. Fu, A. Faust, G. C. H. E. de Croon, and V. J. Reddi, "Tiny Robot Learning (tinyRL) for Source Seeking on a Nano Quadcopter," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, no. Icra, 2021, pp. 7242–7248.

[30] A. Briod, J.-C. Zufferey, and D. Floreano, "Optic-Flow Based Control of a 46g Quadrotor," in *Workshop on Vision-based Closed-Loop Control and Navigation of Micro Helicopters in GPS-denied Environments, IROS 2013*, 2013.

[31] R. J. Moore, K. Dantu, G. L. Barrows, and R. Nagpal, "Autonomous MAV guidance with a lightweight omnidirectional vision sensor," in *Proceedings - IEEE International Conference on Robotics and Automation*, sep 2014, pp. 3856–3861.

[32] K. McGuire, G. C. H. E. de Croon, C. De Wagter, K. Tuyls, and H. Kappen, "Efficient Optical Flow and Stereo Vision for Velocity Estimation and Obstacle Avoidance on an Autonomous Pocket Drone," in *IEEE Robotics and Automation Letters*, vol. 2, no. 2, apr 2017, pp. 1070–1076.

[33] O. Dunkley, J. J. Engel, J. Sturm, and D. Cremers, "Visual-Inertial Navigation for a Camera-Equipped 25g Nano-Quadrotor," in *IROS2014 aerial open source robotics workshop*, 2014, p. 2.

[34] F. Candan, A. Beke, and T. Kumbasar, "Design and Deployment of Fuzzy PID Controllers to the nano quadcopter Crazyflie 2.0," in *2018 IEEE (SMC) International Conference on Innovations in Intelligent Systems and Applications, INISTA 2018*, sep 2018.

[35] A. Anwar and A. Raychowdhury, "Autonomous Navigation via Deep Reinforcement Learning for Resource Constraint Edge Nodes Using Transfer Learning," *IEEE Access*, vol. 8, pp. 26 549–26 560, oct 2020.

[36] A. Suleiman, Z. Zhang, L. Carlone, S. Karaman, and V. Sze, "Navion: A 2-mW Fully Integrated Real-Time Visual-Inertial Odometry Accelerator for Autonomous Navigation of Nano Drones," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 4, pp. 1106–1119, apr 2019.

[37] Z. Li, Y. Chen, L. Gong, L. Liu, D. Sylvester, D. Blaauw, and H. S. Kim, "An 879GOPS 243mW 80fps VGA Fully Visual CNN-SLAM Processor for Wide-Range Autonomous Exploration," in *IEEE International Solid-State Circuits Conference*, mar 2019, pp. 134–136.

[38] M. Hosseini and T. Mohsenin, "Binary Precision Neural Network Manycore Accelerator," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 17, no. 2, pp. 1–27, apr 2021.

[39] N. K. Manjunath, A. Shiri, M. Hosseini, B. Prakash, N. R. Waytowich, and T. Mohsenin, "An Energy Efficient EdgeAI Autoencoder Accelerator for Reinforcement Learning," *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 182–195, jan 2021.

[40] D. Palossi, F. Conti, and L. Benini, "An open source and open hardware deep learning-powered visual navigation engine for autonomous nano-UAVs," in *Proceedings - 15th Annual International Conference on Distributed Computing in Sensor Systems, DCOSS 2019*, may 2019, pp. 604–611.

[41] D. Palossi, A. Loquercio, F. Conti, E. Flamand, D. Scaramuzza, and L. Benini, "A 64-mW DNN-Based Visual Navigation Engine for Autonomous Nano-Drones," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8357–8371, may 2019.

[42] D. Palossi, N. Zimmerman, A. Burrello, F. Conti, H. Muller, L. M. Gambardella, L. Benini, A. Giusti, and J. Guzzi, "Fully Onboard AI-Powered Human-Drone Pose Estimation on Ultralow-Power Autonomous Flying Nano-UAVs," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1913–1929, feb 2022.

[43] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 325–341.

[44] M. Zhai, X. Xiang, R. Zhang, N. Lv, and A. E. Saddik, "Ad-net: Attention Guided Network for Optical Flow Estimation Using Dilated Convolution," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, may 2019, pp. 2207–2211.

[45] G. Zuo, C. Zhang, J. Tong, D. Gong, and M. You, "Edge Detection-Based Optical Flow Estimation Method," in *2021 IEEE 11th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems, CYBER 2021*, jul 2021, pp. 873–878.

[46] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, aug 2020.

[47] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, oct 2017, pp. 1800–1807.

[48] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," apr 2017.

[49] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, jul 2018, pp. 6848–6856.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, jan 2018, pp. 4510–4520.

[51] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, "BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs," *CVPR Workshop on Computer Vision for Augmented and Virtual Reality*, jul 2019.

[52] E. Ilg, T. Saikia, M. Keuper, and T. Brox, "Occlusions, Motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 614–630.

[53] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, dec 2015.

[54] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, dec 2016, pp. 4040–4048.

[55] K. Souhila and A. Karim, "Optical flow based robot obstacle avoidance," *International Journal of Advanced Robotic Systems*, vol. 4, no. 1, p. 2, mar 2007.

[56] G. Cho, J. Kim, and H. Oh, "Vision-based obstacle avoidance strategies for MAVs using optical flows in 3-D textured environments," *Sensors*, vol. 19, no. 11, p. 2523, jun 2019.

[57] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012, pp. 611–625.