

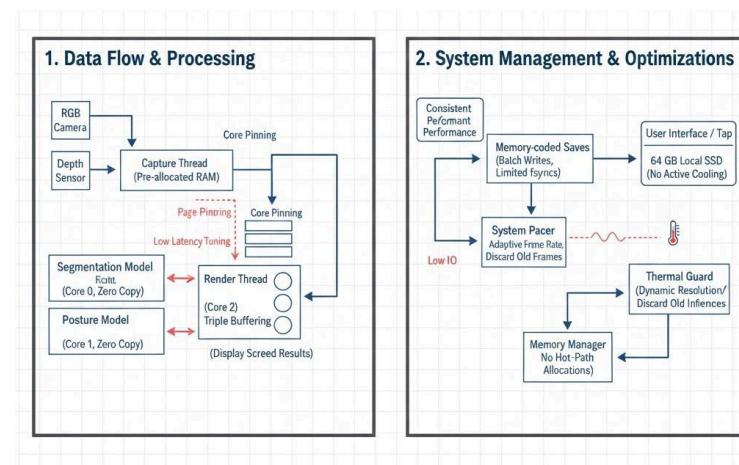
Introduction :

More recent developments in edge artificial intelligence (AI) have made interactive and real-time computing applications such as Ai-generated fashion mirrors which shows clothing fit using on device computer vision .. This system has computer vision ,deep learning and real-time rendering with strict latency and power .This research examines one of the commercial AI fashion mirror kiosks, installed in shopping malls. A quad-core ARM Cortex-A76 processor, a limited cache, and 6 GB LPDDR4 RAM, passive cooling, and a local SSD make up each kiosk. It is a live body segmentation system, a posture detector, a real-time cloth overlay generator (2-3 frames per second) and optional generation outfits are saved locally.

Background/Related Work

The existing literature on edge AI systems reveals that there are major constraints in performance. Sze et al. (2017) present the idea of minimizing energy and latency by reducing main-memory transfers by means of data reuse and efficient DNN design but their research presupposes using specialized accelerators instead of general-purpose ARM processors. Mittal (2016) discusses the problem of memory contention through cache partitioning and finite queues in an effort to enhance predictability but not considering thermal constraints in fanless hardware. Patterson and Hennessy (2021) stress that processing must be resident in the cache to be efficient, whereas Meza et al. (2015) demonstrate that the large number of small writes will accelerate the wear of SSD. These designs were evolved to efficiency-oriented design to reliability concerns but address problems independent of each other, creating a gap of a combined, real-time edge pipeline design.

Methodology / System Design



This is a low-latency pipeline which is optimized on quad-core ARM using pre-allocated memory pools, zero-copy access and triple buffering to remove bottlenecks. Hardware tuning Hardware level tuning, such as core pinning and page coloring, reduces interprocess cache contention. To maintain SSD life disk I/O can only be written in batches caused by a user. Lastly, the thermal guards adaptively regulate the resolution and frame rates, which maintain good performance without causing hardware throttling and overheating when used.

Given: Frame size = 3 MB, Save size = 10 MB, SSD endurance = 100 TBW (102,400 GB)

Write Strategy	Daily Writes	Calculation	Lifetime	With WAF
Save on tap only (300 taps/day)	$300 \times 10 \text{ MB}$ $= 2.93 \text{ GB}$	$102,400 \div 2.93$	95.8 years	47.9 years (WAF=2)
Dump frames to disk (2.5 fps, 8 hrs/day)	$2.5 \times 3 \text{ MB} \times 8 \text{ hrs}$ $= 216 \text{ GB}$	$102,400 \div 216$	1.3 years	5 months (WAF=3)
Compressed thumbnails (500/day, batched)	$500 \times 0.3 \text{ MB}$ $= 0.15 \text{ GB}$	$102,400 \div 0.3$	935 years	467 years (WAF=2)

The continuous frame-dumping strategy causes SSD failure in moth , on the other hand tap based or compressed thumbnail provides decades of use .

Discussion:

Lag and flicker are the results of memory contention or heating slowing down AI processing to the extent that it becomes visibly stuttering. By using small-size queues to drop outdated frames, system freezing is avoided and a smooth flow is ensured. Fanless devices need thermal management, therefore, dynamically decreasing frame rates and resolution would provide a predictable performance. Morally, local processing that is characterized by deletion of data and localized counting improves privacy. Nevertheless, with hardware constraints, software-based memory isolation and frame smoothing can be required to manage heavy compression and user occlusions.

Conclusion :

The pilot issues are owing to bad memory management and too many disk writes. The choice of ring-buffer with small queues balances the performance, whereas the compression of the storage stream to which the data are written only when users save removes almost all the drive wear. Speed is maintained when the heat is controlled. These patches enhance user experience, reduce maintenance expenses and facilitate sustainable and privacy friendly AI in shops. The remedy solves all fundamental problems: lag, flickering, overheating and untimely hardware breakdown.

References

Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *arXiv*.

<https://doi.org/10.48550/arxiv.1510.00149>

Jayaseelan, R., Mitra, T., & Li, X. (2009). Estimating the worst-case energy consumption of embedded software. *Proceedings of the 30th IEEE Real-Time Systems Symposium*, 81–90.

<https://doi.org/10.1109/RTSS.2009.43>

Liu, C. L., & Layland, J. W. (1973). Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM*, 20(1), 46–61.

<https://doi.org/10.1145/321738.321743>

Meza, J., Wu, Q., Kumar, S., & Mutlu, O. (2015). A large-scale study of flash memory failures in the field. *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, 177–190. <https://doi.org/10.1145/2745844.2745848>

Mittal, S. (2016). A survey of cache bypassing techniques. *Journal of Low Power Electronics and Applications*, 6(2), 5. <https://doi.org/10.3390/jlpea6020005>

Patterson, D. A., & Hennessy, J. L. (2021). *Computer organization and design: The hardware/software interface* (6th ed.). Morgan Kaufmann.

Sze, V., Chen, Y.-H., Yang, T.-J., & Emer, J. S. (2017). Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12), 2295–2329.

<https://doi.org/10.1109/JPROC.2017.2761740>