STAR-NT: Spatiotemporal Acceleration of Real-Time Neural Transparency Rendering

Grigoris Tsopouridis[1][0000−0001−8033−5481], Christos Georgiou-Mousses[1][0009−0004−4912−3332], Aris Panagiotidis[1][0009−0008−2964−4359], Andreas Vasilakis[2][0000−0001−6895−3324] David Corrigan[3], Tobias A. Franke[3][0009−0003−9025−9277], Aleksei Gorbonosov[3], Andrei Astapov[3], and Ioannis Fudos[1][0000−0002−4137−0986]

[1] University of Ioannina, Greece {g.thopouridis,fudos}@uoi.gr
[2] Athens University of Economics and Business, Greece
[3] Huawei Ireland Research Center, Ireland

Abstract. Neural order-independent transparency delivers high-quality rendering of overlapping transparent surfaces, but its geometry passes and network input generation remain costly, particularly on mobile and legacy hardware. We present a spatiotemporal acceleration framework that exploits spatial and temporal coherence to reduce this overhead while preserving visual quality. Spatially, we use adaptive quadtree-based screen-space subdivision to scale geometry pass resolution according to local color variance. Temporally, selected frames reuse the previous transparency result through depth-based reprojection instead of full rendering. Together, these optimizations reduce rendering cost and integrate efficiently into existing real-time rendering pipelines.

Keywords: Order Independent Transparency · Rendering · Visibility Determination· Neural Networks · Neural Shading · Deep Learning · Neural Rendering · GPU Performance

1 Introduction

Order-Independent Transparency (OIT) is a fundamental challenge in real-time computer graphics that enables correct rendering of transparent objects without depth-based sorting. Since correct alpha blending [15] requires sorting, it is impractical for complex intersecting geometry, and traditional object-sorting approaches may produce artifacts under multiple overlapping transparent surfaces. OIT avoids these limitations by compositing transparency independently of rendering order, improving visual realism in games, simulations, and visual effects.

Traditional OIT methods are typically divided into *exact* and *approximate* approaches. Exact methods, such as depth peeling [7, 3] and the A-buffer [5, 22], correctly capture all fragments per pixel but incur high and scene-dependent memory and performance costs, making them unsuitable for real-time applications. Approximate methods, including Weighted Blended OIT [11] and Moment-

Based transparency [13, 16], provide bounded cost and predictable performance, but introduce visual artifacts due to compositing approximations.

Recent work has explored neural OIT, where learned models approximate transparency compositing with bounded resources. DFAOIT [20] uses two depth-peeled layers and per-pixel statistics to predict final pixel colors via a lightweight neural network. While achieving high visual quality with constant memory usage, it incurs significant overhead from per-pixel inference and input generation, limiting performance on low-end and legacy hardware.

In this work, we present a dual-domain acceleration framework for neural OIT that reduces computational overhead through spatial and temporal optimizations. Our key insight is that geometry pass cost can be reduced by exploiting (i) spatial redundancy, where low-complexity may not require full-resolution evaluation, (ii) temporal coherence, where consecutive frames share structure enabling transparency reprojection, and (iii) a streamlined network design that predicts only tail transparency color for improved efficiency. This paper makes the following technical contributions:

- A quadtree-based screen-space adaptive subdivision scheme that uses local color variance to adjust geometry pass resolution, reducing shading and overdraw. We additionally employ depth-based temporal reprojection to reuse previous transparency results via frame-to-frame coherence.
- A redesigned neural network with a reduced input feature set, addressing limitations of DFAOIT [20] while improving inference efficiency and maintaining comparable or better quality than non-neural OIT methods.
- A unified framework combining spatial and temporal optimizations, achieving an average 39% speedup while preserving order-independence and requiring no scene-specific training or preprocessing.

Our approach preserves order-independence and requires no preprocessing, scene-specific training, or modification to the underlying OIT method. Operating at the rendering-pipeline level, it can be integrated into existing neural transparency techniques, improving their suitability for interactive applications such as medical visualization, cultural heritage, and mobile rendering.

2 Related Work

Multifragment Rendering. Early OIT methods focused on exact solutions. The A-Buffer [5] stores and sorts all per-pixel fragments before alpha compositing [15], achieving correct results at the cost of unbounded memory. Depth peeling [3] extracts and blends layers iteratively, but its cost scales with the number of visible layers. Bounded variants such as the k-buffer [2] limit per-pixel storage for predictable performance, trading accuracy for efficiency. Despite these improvements, exact multifragment methods remain costly in scenes with high depth complexity.

Blended Transparency. Approximate OIT methods reduce the memory and computational costs of exact approaches by avoiding fragment sorting and

multi-pass processing. Weighted sum [12] and weighted average [3] accumulate fragment contributions with a simple color weighted sum or average, achieving high performance at the expense of visual artifacts. Weighted Blended OIT (WBOIT) [11] improves quality using empirical, user-defined depth and opacity-based weights, but its heuristic formulation can produce artifacts in scenes with high opacity variation or depth complexity. Layered WBOIT [8] further improves quality by applying blending per depth bin, at increased computational cost.

Hybrid Transparency. Hybrid transparency combines exact and approximate OIT to balance visual accuracy and computational efficiency. These methods typically compute a small set of foreground layers exactly while approximating the remaining fragments. Maule et al. [10] use a k-buffer for the foremost fragments and approximate the tail using weighted blending. This improves accuracy for visually important layers while avoiding the unbounded memory cost of A-buffer and the high overhead of depth peeling. However, performance and quality remain sensitive to the chosen number of exact layers.

Neural Transparency. Recently, a number of methods have explored the use of neural networks to enhance or replace the conventional transparency pipeline [18]. Deep and Fast Approximate OIT [20] uses a lightweight neural network to predict the final composite per-pixel colors from their color and opacity statistics, allowing bounded-cost per-pixel inference suitable for real-time use. Neural Moment Transparency [19] follows a hybrid strategy that improves quality over moment-based transparency [13] by replacing selected stages with learned models, but presents a higher computational cost due to additional moment computation and reconstruction steps. Deep Hybrid Order-Independent Transparency [17] integrates a neural pixel-importance estimator into a variable k-buffer pipeline, achieving higher quality at the cost of significantly increased computational and memory demands. Neural networks have also been used to approximate screen-space effects such as ambient occlusion, depth of field, indirect illumination [14], and shadow mapping [6], achieving high-quality results at reduced computational cost.

Accelerating Rendering. Recent rendering advances use tile-based processing and coarse pixel shading [21] to improve performance under bandwidth and power constraints. Tile-based methods partition the screen into spatial regions to improve cache efficiency and reduce memory traffic, lowering shading rates in the GPU pipeline based on the tile resolution. Temporal methods such as TAA [9] and neural frame interpolation [4] exploit frame-to-frame coherence to reduce computation. We apply the same principles to accelerate neural OIT while preserving visual quality.

3 STAR-NT: Spatial-Temporal Acceleration for Real-time Neural Transparency

We present an acceleration framework for neural transparency rendering that combines adaptive resolution control with temporal frame interpolation. An adaptive screen-space tile subdivision scheme, guided by a transparency color dif-

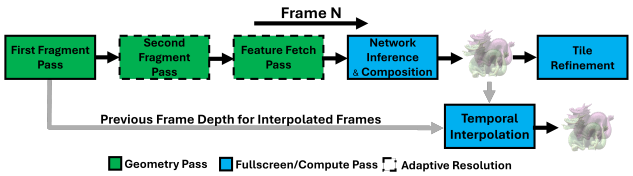

Fig. 1. Geometry passes execute at adaptively reduced resolutions while inference and composition run at full resolution, driving tile refinement for the subsequent frame. First fragment depth is reused for temporal reprojection (grey arrow).

ference metric, dynamically adjusts the resolution of geometry passes, reducing rendering cost while preserving visual fidelity. Tiles computed in frame F_t guide the adaptive resolution of frame F_{t+1}. In addition, we utilize a reprojection-based temporal interpolation stage that exploits temporal coherence under smooth camera motion to amortize computation across frames. Finally, we refine the neural network input features to improve performance and address some of the limitations of DFAOIT [20] (See supplemental material). We build upon the DFAOIT low-end version, using three geometry passes (Fig. 1), avoiding interlocks, making this method suitable for mobile and legacy hardware. The first two passes compute the first and the second closest fragments, while the third pass computes the input features of our neural network. We utilize compute-shader-based tiled processing, mixed-resolution rendering, achieving high-quality transparency rendering with a GPU memory footprint comparable to DFAOIT (75 MB at 1080p). Shader source code is available online: https://github.com/gtsopus/STAR-NT.

3.1 Adaptive Screen-space Tiles

The rendering cost of neural transparency is dominated by geometry passes that rasterize the full scene multiple times. This cost is highly non-uniform across the screen as regions with dense, overlapping, or high-contrast transparent geometry demand greater fidelity than flat or sparsely covered areas. Rather than rendering all passes at full resolution, we employ an adaptive, bitmask-based screen-space tile subdivision system that works as a quality metric to selectively drive the resolution of each geometry pass, based on the corner-sample color variation measure D_i (See supplemental material).

The proposed adaptive tiling scheme dynamically subdivides the screen into variable-resolution regions according to image-space color variation measured from the neural OIT inference output. Starting from 16×16 base tiles, the method constructs a four-level quadtree hierarchy down to 2×2 regions, where each tile encodes its refinement state compactly within a 21-bit mask stored in a single 32-bit integer. Refinement is evaluated in a post-transparency compute pass using only the four corner samples of each node, with local variation estimated through the mean pairwise Euclidean color distance D_i. This constant-cost $O(1)$ evaluation enables efficient split and merge decisions driven by thresholds, with the split thresholds being smaller than the merge thresholds, reducing

temporal instability while preserving structural consistency through top-down refinement. The resulting hierarchy is reused across frames to guide spatially adaptive rendering resolution, achieving efficient detail allocation while avoiding the bandwidth overhead of full-tile analysis. To reduce perceptual artifacts, neighboring tiles may differ by at most one refinement level, and tiles may change by only one level per frame. Since refinement levels are limited to $\{0, 1, 2, 3\}$, two relaxation iterations ($k = 2$) are sufficient to enforce spatial consistency across the grid.

The resulting tile hierarchy is then used in the subsequent frame. In particular, the number of active tiles at each level serves as a demand indicator for our adaptive transparency rendering (Sec. 3.2), guiding the second and third pass resolutions.

3.2 Adaptive Tile-Guided Resolution

The rendering cost of each geometry pass is proportional to the resolution at which it is executed. For transparency, we observe that the visual contribution of each successive pass diminishes under the Porter-Duff over-compositing operator [15], as the color contribution of layer p is attenuated by the accumulated transmittance of all preceding layers,

$$C_{\text{out}} = \sum_{p=1}^{P} C_p \cdot \prod_{q=1}^{p-1} (1 - \alpha_q), \tag{1}$$

where C_p and α_p are the color and opacity of layer p respectively. As depth increases, the product $\prod_q (1 - \alpha_q)$ decreases, attenuating the contribution of deeper layers to the final result. This motivates rendering successive passes at progressively reduced resolution, as the loss in spatial fidelity is masked by the reduced transmittance of the affected layers.

The first pass is always rendered at full resolution since it captures the foremost transparent layer, which contributes most strongly to the final image and contains the most perceptually important detail. Reducing its resolution introduces visible artifacts that later passes cannot recover.

Passes 2 and 3 are rendered at resolutions determined by the scene complexity index $\bar{w}$, computed from the tile level histogram of the previous frame (Section 3.1). The index is defined as the level-weighted mean of the occupied tile distribution: $\bar{w} = \frac{\sum_{\ell=0}^{3} \ell \cdot N_\ell}{\sum_{\ell=0}^{3} N_\ell}$, where N_ℓ denotes the number of occupied tiles at level ℓ. The index $\bar{w} \in [0, 3]$ varies continuously as the tile distribution shifts along scene and camera movements. Each pass maps $\bar{w}$ to a render resolution scale through a pass-specific exponential function:

$$s_p = \text{clamp}\left(2^{\bar{w} - \delta_p},\ 0.25,\ 1.0\right), \delta_2 = 2.2,\ \delta_3 = 3.0 \tag{2}$$

The base-2 exponential form is a natural choice given the power-of-two structure of the subdivision hierarchy: in the unclamped range, a unit increase in $\bar{w}$ corresponds to a doubling of the render scale. The offset δ_p sets the crossover

point at which pass p attains full resolution: $s_2 = 1$ when $\bar{w} = 2.2$, and $s_3 = 1$ when $\bar{w} = 3.0$. Therefore, δ_p determines the trade-off between performance and quality. Pass 3 is assigned a higher offset, rendering it at lower resolution than Pass 2 under equivalent complexity, as its tail color statistics are used by the neural network, which can compensate for partially degraded inputs. Both scales are clamped to a minimum of 0.25, below which quality degrades unacceptably with diminishing performance returns. Pass 3 resolution changes may cause minor temporal artifacts, but hysteresis in split/merge thresholds prevents oscillations, and large changes only occur during major scene or camera shifts.

Since neural network inference operates at full resolution, the outputs of Passes 2 and 3 are first upscaled to framebuffer resolution using bilinear interpolation. The upscaled Pass 3 buffers are then provided as neural network inputs (Sec. 3.3) and composited together with the full-resolution Pass 1 output, the upscaled Pass 2 result, and the background color.

3.3 Neural Network Improvements

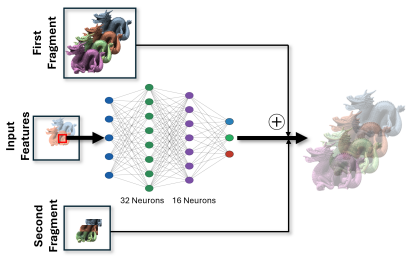

Fig. 2. Our neural network predicts the transparency tail color using an adaptive resolution feature pass. The tail color is then blended with the first two fragments, compositing the final OIT color.

We build upon the low-end hardware variant of DFAOIT [20], which uses depth peeling to extract the two nearest fragments and an additional geometry pass to accumulate per-pixel statistics without pixel synchronization, making it suitable for mobile and legacy hardware.

Our neural network is a compact MLP with two hidden layers (32 and 16 neurons) and a 3-neuron ReLU output predicting the tail RGB color. Following DFAOIT [20], the architecture differs only in the output activation (ReLU instead of sigmoid) and predicts only the transparent tail rather than the full transparency color. The network is trained offline on 10 million transparent samples using MSE loss and Adam.

The original DFAOIT [20] directly predicts the transparency color C_t using 10 input features, related to pixel opacity and color:

$$C_p = C_t + C_b \prod_{i=1}^{n} (1 - \alpha_i). \tag{3}$$

The 10 features are: the exact Porter-Duff blended color of the two front fragments, the average color C_{avg} and average opacity α_{avg} of the remaining $n-2$ fragments, the accumulated premultiplied color C_{acc}, and the fragment count n.

We address two limitations of DFAOIT: mixing the exact contribution of the first two depth-peeled fragments with the approximate tail contribution, and the use of unnormalized C_{acc}, which can cause color artifacts. To overcome these issues, the network predicts only the *tail color*, decoupling the exact two-fragment term from the learned approximation:

$$C_p = C_{\text{exact},2} + T_2 \cdot f(\mathbf{x}) + \prod_{i=1}^{n} (1 - a_i) C_b \tag{4}$$

where $C_{\text{exact},2} = \alpha_1 C_1 + (1 - \alpha_1)\alpha_2 C_2$ is the exact two-layer composite, $T_2 = (1 - \alpha_1)(1 - \alpha_2)$ is the transmittance through the two peeled layers, $f(x)$ is the inferred tail color, and $\prod_{i=1}^{n}(1 - a_i)$ is the transmittance of all transparent fragments.

The network utilizes a five-float input vector:

- Total fragment count: N.
- Tail transmittance: $t_{\text{tail}} = \alpha_{\text{acc}}/T_2$.
- Normalized premultiplied RGB color of the tail fragments: $\tilde{\mathbf{C}}_{\text{pm}} = C_{\text{acc}}/\alpha_{\text{acc}}$, where $C_{\text{acc}} = \sum_{i=3}^{N} \alpha_i \mathbf{C}_i$ is the premultiplied color accumulated over the tail fragments.

The reduction from 10 to 5 features decreases weight tensor memory and inference cost, improving suitability for constrained hardware (Tab. 1), while tail normalization addresses the color artifacts identified in the original method (See supplemental material).

Geometry passes dominate runtime cost. To reduce this overhead under smooth camera motion, we optionally reuse previous frames through reprojection-based interpolation. Frames alternate between *real renders*, where the full pipeline is executed, and *interpolated frames*, where geometry passes and network inference are skipped. The interpolation interval is adapted at runtime according to the measured frame rate and constrained to a maximum of four frames. When performance drops below the target frame rate, the method reverts to standard rendering without temporal interpolation. Interpolated frames are generated by reprojecting pixels using the nearest available depth and the previous view-projection matrix. Similar to common real-time rendering approaches, this lightweight optimization prioritizes performance and may degrade under rapid motion or highly dynamic transparent geometry.

4 Experimental Evaluation

We evaluate STAR-NT against DFAOIT [20] and WBOIT [11] on ten scenes with varying depth complexity D (up to 136 fragments per pixel), color, and opacity, ranging from low to high complexity. WBOIT is used as the baseline OIT method due to its strong real-time performance and plausible quality, using the original authors recommended weight function [11]. Experiments are run at 1920×1080 on three representative platforms: Huawei Mate 70 Pro (mobile), NVIDIA GTX 1660 Super (legacy desktop), and NVIDIA RTX 5080 (high-end desktop). Image quality is evaluated using MSE and FLIP [1], where lower values indicate closer agreement with the A-buffer ground truth.

4.1 Quality

Figure 3 and the supplementary material report MSE and FLIP errors for all scenes against the A-buffer [22], which serves as ground truth. STAR-NT achieves lower error than WBOIT on both metrics, with average MSE and FLIP reductions of 32.2% and 30.2%, respectively. DFAOIT achieves slightly lower error than STAR-NT in most scenes, as STAR-NT uses reduced-resolution geometry passes.

Hairball and Powerplant represent challenging high-depth-complexity cases (Supplementary Material). In Hairball, WBOIT achieves the lowest error due to the extremely large fragment count, where STAR-NTs tail prediction operates on more heavily aggregated statistics, while WBOITs weighted blending is better suited for this scene. DFAOIT exhibits visible color artifacts, which are mitigated by our revised inputs. In Powerplant ($D = 96$), STAR-NT achieves the best overall quality, while DFAOIT performs worse than WBOIT, confirming that the improved input representation and decoupled compositing are particularly beneficial in these scenes.

Excluding these two scenes, STAR-NT achieves 40% and 42% lower FLIP and MSE, respectively, relative to WBOIT, compared to 50% and 53% for DFAOIT. Despite operating with reduced-resolution geometry passes and a smaller feature set, STAR-NT maintains competitive quality while mitigating the color artifacts and instability observed in DFAOIT for high-depth-complexity scenes. Minor artifacts mainly stem from downsampled inputs. Likewise, fast motion and disocclusions are well-known limitations of temporal reconstruction methods. Since the temporal interpolation stage is modular, it can be replaced with more advanced schemes.

We evaluate our method on three platforms and ten scenes with varying depth complexity. STAR-NT achieves consistent frame-time reductions over DFAOIT, with an average speedup of 39%, with the largest gains on legacy hardware where geometry-pass cost dominates (Tab. 2). STAR-NT matches the runtime of WBOIT, a single-pass non-neural baseline, while maintaining substantially higher image quality, making neural transparency rendering competitive in performance without sacrificing its quality advantages.

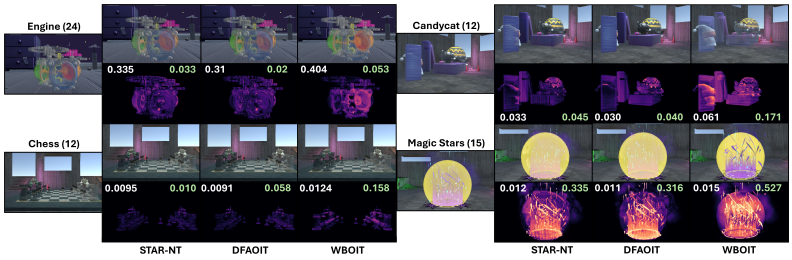

Fig. 3. Quality comparison of methods: $FLIP$ mean error (white) and $MSE * 10^2$ (green). The ground truth is shown on the left with the depth complexity (maximum number of transparent layers) in the parenthesis.

Scene (P1%,P2%,P3%, D)	Pass 1	Pass 2	Pass 2↓	Pass 3	Pass 3↓	Compute	Inf. DFAOIT	Inf. STAR	Total DFAOIT	Total STAR-NT w/o Temporal	Total STAR	Speedup
Smoke (100,50,50, 56)	0.042	0.068	0.024	0.138	0.038	0.052	0.114	0.086	0.362	0.242	0.194	46.27%
Hairball (100,50,50, 136)	0.425	0.559	0.323	1.000	0.415	0.100	0.160	0.068	2.144	1.331	0.770	64.08%
Candycat (100,50,25, 12)	0.048	0.040	0.025	0.089	0.020	0.071	0.102	0.078	0.279	0.242	0.210	24.73%

Table 1. Ablation and per-pass GPU timings (ms) for three scenes at varying adaptive resolution scales (P1%/P2%/P3%). Pass N↓ denotes reduced-resolution STAR passes. Compute includes tile evaluation and upsampling. Inference reports neural network inference time for the original and redesigned networks. All timings are in ms, D denotes maximum depth complexity.

Improvement Breakdown Table 1 shows a per-pass timing breakdown for low (Candycat), medium (Smoke), and high (Hairball) complexity scenes on an RTX 5080. The main improvement comes from adaptive resolution scaling in geometry Pass 2 and Pass 3, with the largest gains in Hairball and Smoke due to high geometry cost and overdraw. The redesigned network inputs (Sec. 3.3) provide an additional 35% inference speedup by reducing input features from 10 to 5. Temporal interpolation further amortizes full geometry passes across frames.

Desktop Performance Table 2 reports frame times on the RTX 5080. STAR-NT achieves an average 36.8% reduction over DFAOIT across all scenes, with the largest gains in geometry-heavy workloads. Relative to WBOIT, STAR-NT is only 4.4% slower on average while providing substantially higher image quality (Sec. 4.1).

Mobile Performance Table 2 reports frame times on the Huawei Mate 70 Pro. STAR-NT reduces frame time over DFAOIT by an average of 28%, with the largest gains in geometry-heavy scenes such as Tree and Smoke. Powerplant and Hairball show smaller improvements as temporal interpolation is not active due to low frame rates. Relative to WBOIT, STAR-NT is faster in six out of ten scenes, with a 5.5% average advantage. WBOIT remains faster in low-depth-complexity scenes due to its single-pass design, which is difficult to match with a multi-stage pipeline.

Legacy Desktop Performance The GTX 1660 Super results demonstrate an average frame time reduction of 54.0% relative to DFAOIT. The gains are consistent across all scenes and largest in high depth complexity scenes: Hairball and Powerplant see reductions exceeding 60% and 64% respectively. Relative to WBOIT, STAR-NT shows a similar performance, only 0.5% faster, effectively matching a single-pass approximate method while delivering substantially better image quality.

Scene (D)	DFAOIT			STAR-NT			WBOIT		
	Mobile	*Legacy*	*Desktop*	*Mobile*	*Legacy*	*Desktop*	*Mobile*	*Legacy*	*Desktop*
Engine (24)	16.7	3.3	0.8	11.1	1.56	0.54	12.5	1.6	0.58
Hair (80)	11.8	2.6	0.78	9.1	1.44	0.57	8.5	1.2	0.44
Magic Stars (15)	14.1	3.8	1.07	11.2	1.7	0.57	11.4	1.9	0.61
Chess (12)	16.7	4.2	0.9	10.5	1.75	0.6	13.9	1.9	0.52
Clothes (13)	13.9	2.4	0.75	9.1	1.4	0.57	7.7	1.2	0.5
Tree (58)	23.3	3.9	0.85	11.8	1.75	0.68	17.5	1.9	0.65
Smoke (56)	14.9	3.0	-	8.5	1.51	-	12.5	1.4	-
Candycat (12)	13.3	2.5	-	10.0	1.45	-	8.3	1.24	-
Powerplant (98)	55.6	12.3	3.45	52.6	4.4	1.1	47.6	4.2	1.25
Hairball (136)	58.8	11.3	-	47.6	2.7	-	52.6	4.7	-

Table 2. Frame times (ms) on Mobile (Huawei Mate 70 Pro), Legacy (NVIDIA GTX 1660 Super), Desktop (RTX 5080), for scenes of varying maximum depth complexity D.

4.2 Power and Energy Consumption

Table 3 reports frame counts, total power consumption, and energy per frame measured on the Huawei Mate 70 Pro across three representative scenes. We used Huawei SmartPerf to measure current draw in milliamps (mA) under sustained rendering, with all methods running at identical scene, display, and battery conditions.

Total current draw or power is not an indicative energy efficiency metric when methods produce different frame counts; energy per frame provides a fair basis for comparison. The energy in mJ for a frame is given by $EN = I \times F_t \times V_d$, where I is the current in mA, F_t is the frame duration and V_d is the device voltage. STAR-NT consistently consumes less energy per frame than DFAOIT across all three scenes. Relative to WBOIT, STAR-NT consumes less energy per frame in higher-complexity scenes where neural inference overhead is outweighed by the reduction in geometry pass cost, while drawing slightly more in simpler scenes and delivering substantially better image quality (Section 4.1).

To jointly assess image quality and energy efficiency, we define a quality-efficiency metric Q:

$$Q = \frac{E_{\text{random}} - E_{\text{method}}}{EN_{\text{frame}} \times E_{\text{random}}} \times 100 \tag{5}$$

where E_{method} is the per-scene MSE of each method, E_{random} is the MSE of a naive GPU renderer with no OIT, using the "over" operator on random fragment order, and EN_{frame} is the energy consumption per frame in mJ. Q expresses the percentage of improvement versus the naive-blend error (without sorting) gained per mJ of energy consumed. For example if $Q = 1\%$ per mJ for a method this means that each mJ consumed improves the MSE by 1%.

STAR-NT achieves the highest Q in all three measured scenes (Table 3), confirming that its performance gains are not offset by quality degradation. WBOIT scores substantially lower as it offers worse quality and in some cases higher energy per frame. DFAOIT underperforms STAR-NT despite the marginally better raw MSE, due to its higher per-frame energy cost. Overall, STAR-NT delivers the best quality improvement per mJ consumed across all measured scenes.

Scene	Method	FPS	Overall power (mW)	Energy (mJ) per frame	Q % per mJ
Smoke					
	WBOIT	80	7400	92,50	0,25
	DFA	67	8140	121,49	0,36
	STAR-NT	117	8510	72,74	0,55
Candycat					
	WBOIT	120	6845	57,04	0,93
	DFA	75	7215	96,20	0,92
	STAR-NT	100	7400	74,00	1,18
Engine					
	WBOIT	80	6845	85,56	0,68
	DFA	60	7030	117,17	0,64
	STAR-NT	90	7215	80,17	0,93

Table 3. Power consumption and quality-efficiency on the Huawei Mate 70 Pro. Q provides the improvement of quality offered by each mJ consumed.

5 Conclusions

We present STAR-NT, a dual-domain acceleration framework for neural OIT combining adaptive quadtree-based resolution scaling in geometry passes with depth-based temporal reprojection. This reduces spatial and temporal redundancy, yielding an average 39% speedup over DFAOIT and up to 64% on legacy hardware, while requiring no scene-specific preprocessing. The redesigned network inputs improve stability in high-complexity scenes and reduce inference cost. Across platforms, STAR-NT matches the performance of simpler approximate methods while maintaining higher image quality, narrowing the gap between neural transparency and real-time deployment. Future work includes learned frame synthesis, spatially selective inference, and hardware-aware acceleration.

References

1. Andersson, P., Nilsson, J., Akenine-Möller, T., Oskarsson, M., Åström, K., Fairchild, M.D.: Flip: A difference evaluator for alternating images. Proc. ACM Comput. Graph. Interact. Tech. **3**(2) (Aug 2020). https://doi.org/10.1145/3406183
2. Bavoil, L., Callahan, S.P., Lefohn, A., Comba, J.a.L.D., Silva, C.T.: Multi-fragment effects on the gpu using the k-buffer. In: Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games. p. 97–104. I3D '07, ACM, New York, NY, USA (2007). https://doi.org/10.1145/1230100.1230117
3. Bavoil, L., Myers, K.: Order independent transparency with dual depth peeling. In: NVIDIA OpenGL SDK. (2008)
4. Briedis, K.M., Djelouah, A., Meyer, M., McGonigal, I., Gross, M., Schroers, C.: Neural frame interpolation for rendered content. ACM Trans. Graph. **40**(6) (Dec 2021). https://doi.org/10.1145/3478513.3480553

5. Carpenter, L.: The a -buffer, an antialiased hidden surface method. SIGGRAPH Comput. Graph. **18**(3), 103–108 (jan 1984). https://doi.org/10.1145/964965.808585
6. Datta, S., Nowrouzezahrai, D., Schied, C., Dong, Z.: Neural shadow mapping. In: ACM SIGGRAPH 2022 Conference Proceedings. ACM, New York, NY, USA (2022). https://doi.org/10.1145/3528233.3530700
7. Everitt, C.: Interactive order-independent transparency. Tech. rep., Nvidia Corporation (2001)
8. Friederichs, F., Eisemann, M., Eisemann, E.: Layered weighted blended order-independent transparency. In: Graphics Interface. pp. 196–202 (2021)
9. Karis, B.: High quality temporal supersampling. In: ACM SIGGRAPH 2014 Courses. ACM (2014)
10. Maule, M., Comba, J.a., Torchelsen, R., Bastos, R.: Hybrid transparency. In: Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. p. 103–118. ACM, New York, NY, USA (2013). https://doi.org/10.1145/2448196.2448212
11. McGuire, M., Bavoil, L.: Weighted blended order-independent transparency. Journal of Computer Graphics Techniques (JCGT) **2**(2), 122–141 (December 2013)
12. Meshkin, H.: Sort-independent alpha blending. GDC Session (2007)
13. Münstermann, C., Krumpen, S., Klein, R., Peters, C.: Moment-based order-independent transparency. Proc. ACM Comput. Graph. Interact. Tech. **1**(1) (jul 2018). https://doi.org/10.1145/3203206
14. Nalbach, O., Arabadzhiyska, E., Mehta, D., Seidel, H.P., Ritschel, T.: Deep shading: Convolutional neural networks for screen space shading. Computer Graphics Forum **36**(4), 65–78 (2017). https://doi.org/https://doi.org/10.1111/cgf.13225
15. Porter, T., Duff, T.: Compositing digital images **18**(3), 253–259 (Jan 1984). https://doi.org/10.1145/964965.808606
16. Sharpe, B.: Moment transparency. In: Molnar, S. (ed.) Proceedings of the Conference on High-Performance Graphics, HPG 2018, Vancouver, Canada, August 10-12, 2018. pp. 8:1–8:4. ACM (2018). https://doi.org/10.1145/3231578.3231585
17. Tsopouridis, G., Fudos, I., Vasilakis, A.A.: Deep hybrid order-independent transparency. The Visual Computer **38**(9), 3289–3300 (2022). https://doi.org/10.1007/s00371-022-02562-7
18. Tsopouridis, G., Georgiou-Mousses, C., Fudos, I., Corrigan, D., Franke, T.A.: Traditional and Neural Order-Independent Transparency. In: Mantiuk, R., Hildebrandt, K. (eds.) Eurographics 2025 - Tutorials. The Eurographics Association (2025). https://doi.org/10.2312/egt.20251001
19. Tsopouridis, G., Vasilakis, A., Fudos, I.: Neural moment transparency. In: Eurographics Proceedings, short papers (2024). https://doi.org/10.2312/egs.20241029
20. Tsopouridis, G., Vasilakis, A.A., Fudos, I.: Deep and fast approximate order independent transparency. Computer Graphics Forum **43**(6), e15071 (2024). https://doi.org/10.1111/cgf.15071
21. Vaidyanathan, K et al.: Coarse Pixel Shading. In: Wald, I., Ragan-Kelley, J. (eds.) Eurographics/ ACM SIGGRAPH Symposium on High Performance Graphics. The Eurographics Association (2014). https://doi.org/10.2312/hpg.20141089
22. Yang, J.C., Hensley, J., Grün, H., Thibieroz, N.: Real-time concurrent linked list construction on the gpu. Computer Graphics Forum **29**(4), 1297–1304 (2010). https://doi.org/10.1111/j.1467-8659.2010.01725.x