Eurographics Symposium on Parallel Graphics and Visualization '25
G. Reina, S. Rizzi, C. Gueunet
(Editors)

Volume 0 (2025), Number 0

Supplementary Material for Multi-Density Woodcock Tracking

Alper Sahistan[1], Stefan Zellmann[2], Nate Morrical[3], Valerio Pascucci[1], and Ingo Wald[3]
[1]SCI Institute at the University of Utah, Salt Lake City, UT
[2]University of Cologne, Germany
[3]NVIDIA

Abstract

Volume rendering techniques have increasingly transitioned toward Monte Carlo (MC) methods in recent years due to their flexibility and robustness. However, their applications in multi-channel rendering have yet to be thoroughly explored. Moreover, color blending between channels in traditional compositing-based rendering often lacks a physical basis, leading to user confusion. This paper introduces multi-density Woodcock tracking*, a simple and flexible approach to multi-channel volume rendering, leveraging the strengths of MC methods to produce high-fidelity visuals. Our method offers a physically grounded solution for inter-channel color blending and eliminates the need for arbitrary blending functions. We also propose a unified blending modality by generalizing Woodcock's distance tracking method, facilitating seamless integration of alternative blending functions from prior works. Through evaluation across diverse datasets, we demonstrate that our approach maintains real-time interactivity while achieving high-quality visuals by accumulating frames over time.*

CCS Concepts

• ***Computing methodologies*** → ***Ray tracing;*** *Volumetric models;* • ***Human-centered computing*** → ***Scientific visualization;***

1. Introduction

This supplemental document goes into some details and results we discarded from the main paper due to negative results, insignificance, or lack of space. However, we still see merit in communicating what has not been successful. Moreover, we re-report some of the visual results as larger and more detailed images.

2. Method #3: One DDA Traversal over a Cumulative Majorant Grid

We briefly mention this method in the main paper in the results section. However, the method itself and its results are not present in the paper due to being less impactful and requiring an additional step of processing.

After method #2, another *potential* improvement we identify is to reduce all N majorant grid data to a single one. We propose the usage of a cumulative majorant buffer to reduce multiple accesses to memory. This process ultimately tracks a single channel instead of N different ones, potentially identifying a sample faster.

$$L(x,\omega) = \int_{t=0}^{d} p(t)\left[P_{real}(x)\,L_e(x_t,\omega) + P_{null}(x)\,L(x_t,\omega)\right]dt \quad (1)$$

$$L(x,\omega) = \int_{t=0}^{d} p(t)\left[\sum_{n=1}^{N}\left(P(\theta_n,x)\,L_e(x_t,\omega,\theta_n)\right) + P_{null}(x)\,L(x_t,\omega)\right]dt \quad (2)$$

Just like method #1 and method #2, this method approximates the Equation 2; however, we do not differentiate between different channels' collision probabilities (P_n) until we encounter a real collision. So we essentially revert to formulation in Equation 1, but within each occurrence of a real collision, we use P_n to choose which sample to take (or not to take any, i.e. null-collision).

Unlike other techniques mentioned in the paper, this method uses a single majorant grid, where for each macrocell, a sum of N majorants is stored as a cumulative majorant. However, calculating and storing min and max value pairs for each macrocell is still required as we recalculate majorants based on these values. This method mostly coincides with the technique presented in the overlapping volumes chapter of Fong et al.'s Production Volume Rendering course [FWKH17].

Rendering with this method is quite similar to using a single channel and utilizing a regular Woodcock tracker. We iterate over the cumulative majorant grid with DDA traversal and take Woodcock steps within the cells using the majorant value, m_i, for the given cell at index i. The only significant difference to standard Woodcock tracking occurs during the sampling stage, where we test for true collisions by getting the respective scalar values for each of the N channels. These scalars are used to obtain samples' respective opacities (densities) via transfer function lookups. Then, we importance-sample these channels based on their densities. If all of the samples are rejected, it is a null collision, but if one is accepted, we return that sample's color value and stop the traversal.

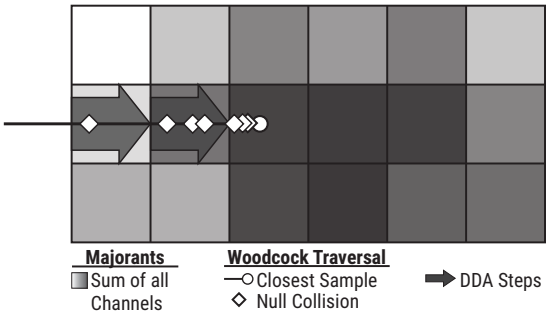


Figure 1: An illustration of the Woodcock tracking process with one DDA traversal over a cumulative majorant grid from a perspective of one ray: A DDA traversal is initiated for the cumulative majorant grid, and between each DDA step, Woodcock steps are taken where null collisions are discarded. These processes are repeated until a sample is found. The sample is accepted, and the mapped color from the transfer function is returned.

Please also refer to the Figure 1 for an illustration and Listing 1 for a kernel implementation.

Listing 1: Multi-channel Woodcock tracking using a DDA traversal over the cumulative majorant grid

```
void cumulativeDDAMultiChannelWoodcock(
                Ray ray, HitRecord& hit, int N){
  DDA dda(ray);
  int cellID = dda.curCell();
  do{//DDA traversal
    float t = dda.cellEntryT();
    float cMaj = majorants[cellID];
    while(true){//Woodcock steps in the cell
      t += woodcockStep(cMaj);
      if(!dda.InCurCell(t))//bounds check
        break;//go to the next cell

      float colThreshold = randFloat()*cMaj;
      //Pick a sample w.r.t density
      for(int n = 0; n < N; n++){
        float scalar = volumeAt(ray.org
                          + ray.dir * t, n);
        float4 sample = trFunc(scalar, n);
        //null collision check for nth channel
        if( sample.w > colThreshold){
          //accept this channel:
          //but discard the "w"
          hit.color = float3(sample);
          hit.t= t;
          dda.stop();//stop the traversal
          break;
        }
        colThreshold -= sample.w;
      }// no channel was selected: null collision
    }cellID = dda.nextCell();
  }while(!dda.shouldStop());
}
```

3. Additional Results

3.1. Performance and Memory Consumption of Method #3

In this section, we re-report some of our benchmarks with method #3. We also report a memory consumption comparison using method #3.

The Figure 2 shows that method #3 is the least performant method among our three implementations of multi-density Woodcock tracking. It can be observed that method #3 reaches its peak performance before the other two methods. In scenarios where traversal depth is much shallower, like the certain angles to Zebrafish or Hurricane, we observe this method to outperform our other implementations for macrocell size is 1^3. However, this case is equivalent to not using a macrocell grid, i.e., DDA traversal over the original grid data.

We can also look at the Table 1 for a comparison of the memory consumption of our methods at their peak performance. As stated before, method #3 reaches peak performance on a much finer macrocell grid compared to method #1 and #2. Due to that fact, this method also consumes more memory while not keeping up with the performance of the other approaches.

In the Figure 3, we observe that even method #3 is generally faster than the ray marching algorithm even while rendering volumetric shadows. However, it also demonstrates poor scaling with the increasing number of channels. As the data structure is a reduced approximation, it stores even less detailed information about the density trend of the volume. This reduction makes the cumulative majorant grid a less adaptive data structure, diminishing the benefits gained from the adaptive stepping of Woodcock tracking. In other words, on average, method #3 behaves much closer to its worst-case of $O(N)$ as the local cumulative majorant is tested against each channel until the threshold is exceeded. In contrast to method #3, our other two approaches prune the redundant traversals much more efficiently, leveraging the occlusion between channels to obtain better average complexity.

3.2. Quality vs. Performance Addendum

The main paper's "similar quality" vs. "similar performance" experiments only included two of our four datasets due to lack of space. This document presents the plots and images generated for three of our large datasets as larger figures. The additional results depicted in Figure 6 support our findings from the original paper where we show Figure 4 and Figure 5.

The Figure 6 draws a similar speed to quality trade-off to Figure 4. The dataset has many dense (active) cells in the viewing direction; therefore, both rendering algorithms quickly complete their traversal and return. For this reason, the inherent space-skipping benefits of a Woodcock tracker are undercut here. However, even if one of the channels were to occupy near-empty cells before the dense cells in the viewing direction, we would have ended up with trade-offs similar to Figure 5. It is common for scientific volumetric data to contain empty cells. For example, CT scans and simulation data often generate boundary cells with uniform values that encase the meaningful features of the dataset.

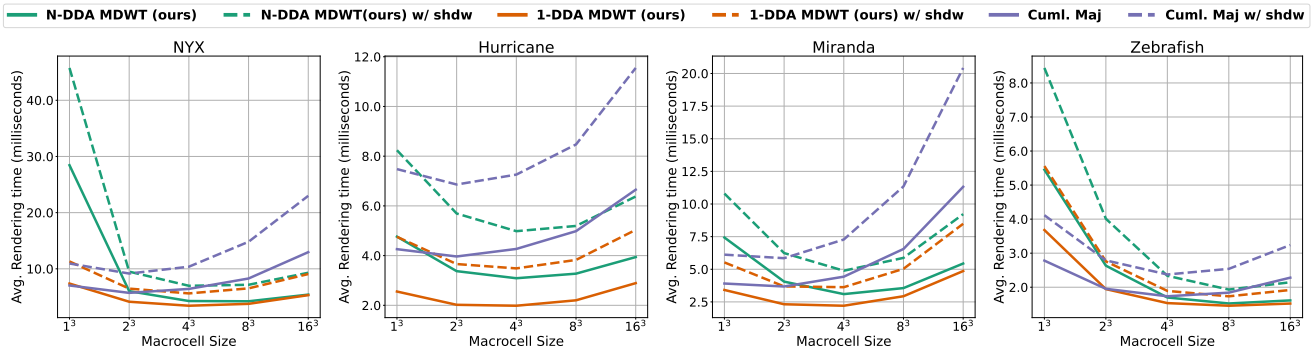


Figure 2: The plots of average rendering times in seconds (lower is better) against increasing macrocell sizes (# of cells within a macrocell) for four datasets. We compare our two methods: method #1, labeled as "N-DDA MDWT (Multi-density Woodcock Tracking)," method #2, labeled as "1-DDA MDWT," and method #3, labeled as "Cuml. Maj". Solid lines are direct volume rendering (DVR), while dashed lines represent DVR with shadows.

Table 1: Theoretical memory consumption for our datasets for the peak performance configurations. Method #1 and #2 use multiple majorant buffers, and method #3 uses a cumulative majorant buffer.

Datasets	# of Chan.	Majorant Config. at Peak Perf.	Data Structure Sizes (MiB)			
			Base	Macrocells	Majorants	Total:
NYX	4	Multiple ($4 \times 128 \times 128 \times 128$)	2048.0	64.0	32.0	2144.0
		Cumulative($1 \times 256 \times 256 \times 256$)		128.0	64.0	2240.0
Hurricane	4	Multiple ($4 \times 125 \times 125 \times 25$)	381.47	11.92	5.96	399.35
		Cumulative ($1 \times 250 \times 250 \times 50$)		23.84	11.92	417.23
Miranda	4	Multiple ($4 \times 96 \times 96 \times 64$)	576.0	18.0	9.0	603.0
		Cumulative ($1 \times 192 \times 192 \times 128$)		36.0	18.0	630.0
Zebrafish	3	Multiple ($3 \times 80 \times 80 \times 15$)	567.19	2.19	1.09	570.46
		Cumulative ($1 \times 160 \times 160 \times 30$)		5.86	2.92	575.97

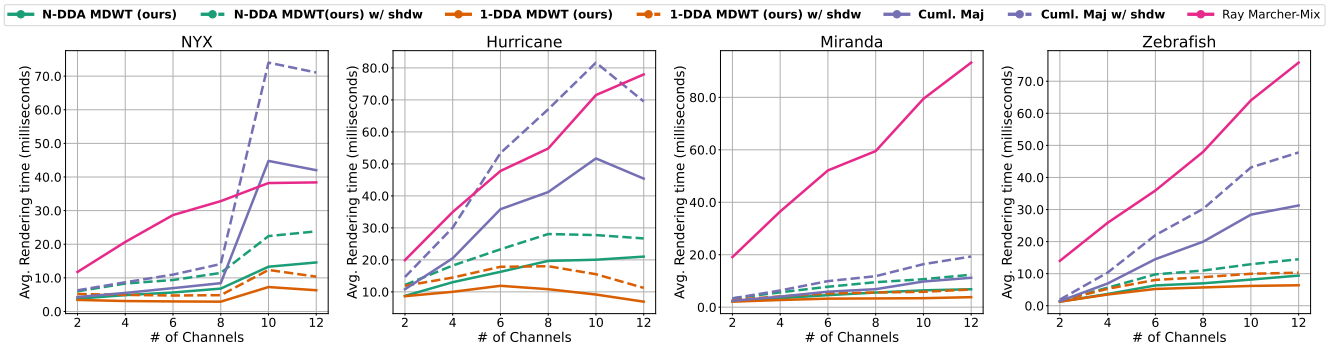


Figure 3: The average rendering times in milliseconds (lower is better) versus the increasing number of channels across four datasets. For each dataset, we present the rendering of individual channels on the left, all channels combined with multi-channel Woodcock tracking on the middle, and a line plot on the right. The line plots compare "N-DDA MDWT (Multi-density Woodcock Tracking)," "1-DDA MDWT" method, "Cuml. Maj" method (#3), and "Ray Marcher-Mix," a ray marcher using mix blending. Dashed lines represent the performance of our methods with volumetric shadows, while solid lines represent others.

3.3. Visual Comparisons for Blending Functions Addendum

We include another omitted result from the paper along with the already reported results from the "Visual Comparisons for Blending Functions" section of the main paper. Table 2 reports visual similarity of an image-space method, two not physically based blending functions, to our physically-based density blending function, which can be considered ground-truth in this case. In addition to providing slightly larger images, we also include the Miranda dataset as a row in this supplementary table.

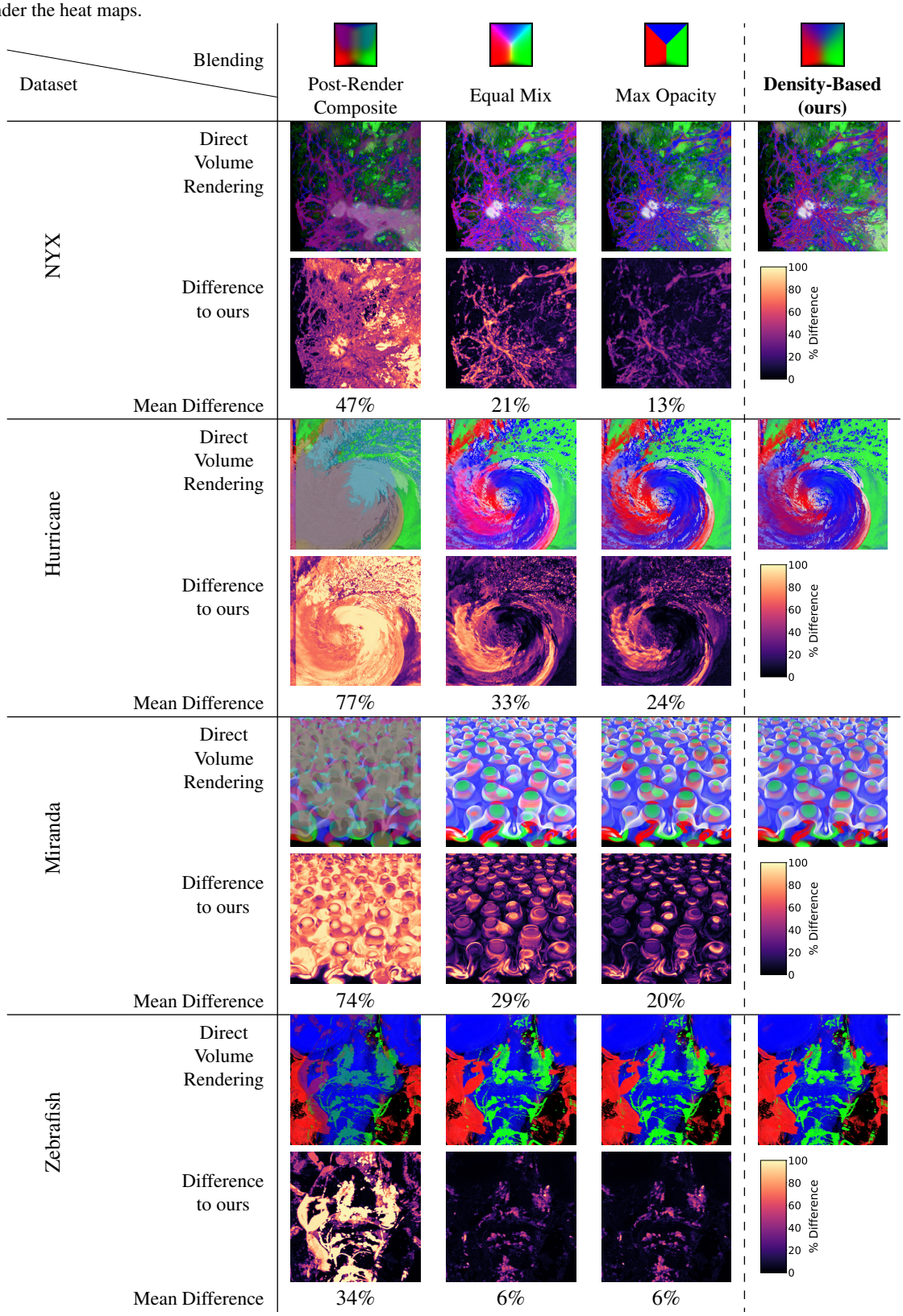
Overall, blending functions behave almost consistently with other results we have (most akin to the Hurricane dataset). For equal mix and max opacity the most difference reported comes from where individual channels overlap with each other.

References

[ANA21] ANDERSSON P., NILSSON J., AKENINE-MÖLLER T.: Visualizing and Communicating Errors in Rendered Images. In *Ray Tracing Gems II*, Marrs A., Shirley P., Wald I., (Eds.). 2021, ch. 19, pp. 301–320. 5

[FWKH17] FONG J., WRENNINGE M., KULLA C., HABEL R.: Production Volume Rendering: SIGGRAPH 2017 Course. In *ACM SIGGRAPH 2017 Courses* (2017), SIGGRAPH '17, p. 51. Section 3.9: Overlapping volumes. doi:10.1145/3084873.3084907. 1

Table 2: Visual comparisons of inter-channel blending functions over test datasets: Direct volume renderings using Emission+Absorption model with identical parameters for various blending functions, accompanied by heat map images depicting the difference to our physically motivated density-based blending function using ꟻLIP metric [ANA21]. The overall mean difference to density-based blending is also reported under the heat maps.



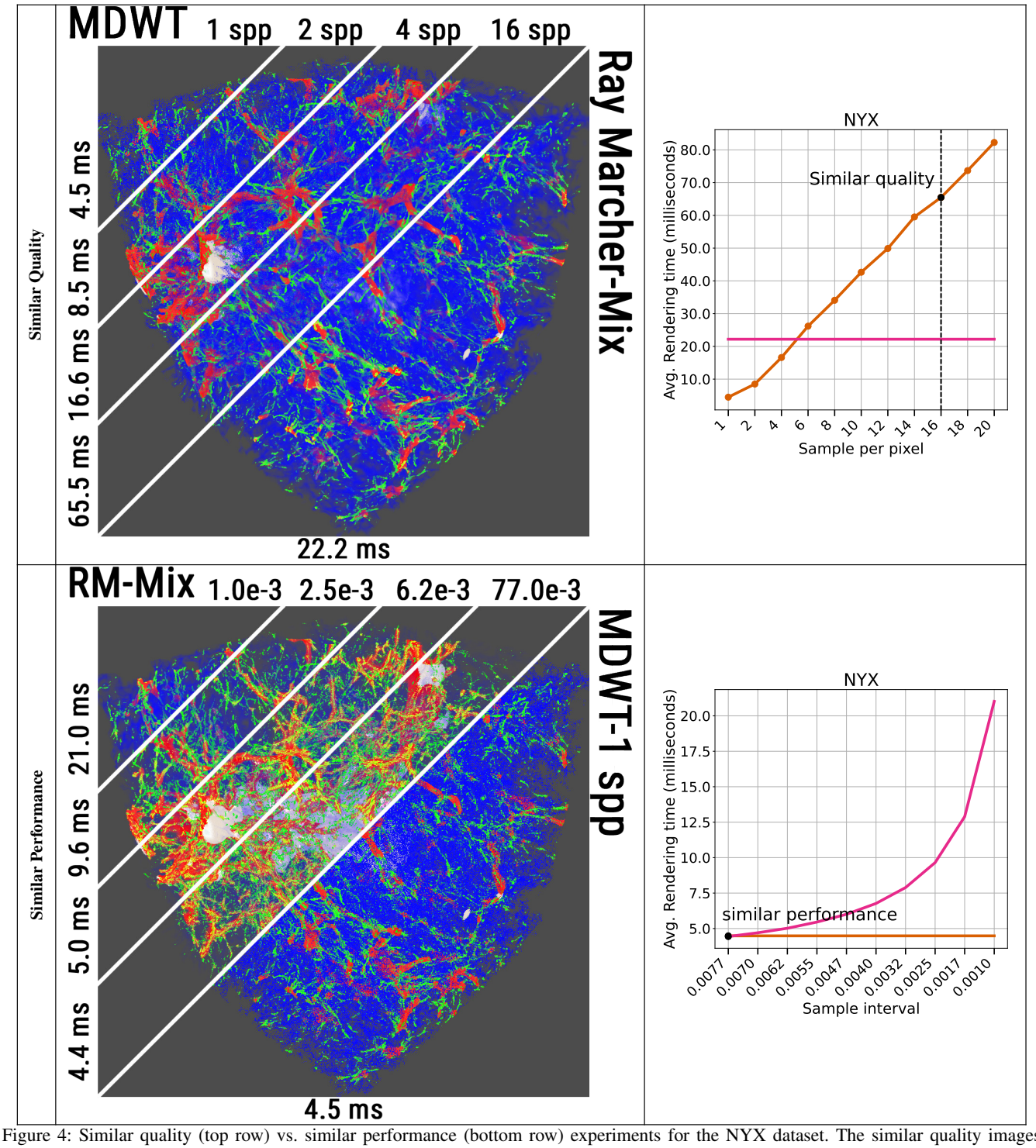


Figure 4: Similar quality (top row) vs. similar performance (bottom row) experiments for the NYX dataset. The similar quality images display slices rendered with our Multi-Density Woodcock Tracking (MDWT) at increasing samples per pixel (spp), while the bottom-right half shows the ray marcher using mix blending with half of the minimum cell size as the sampling interval (Nyquist rate). The similar performance images show slices rendered with the ray marcher and mix blending (RM-Mix) for increasing sampling intervals, with the bottom-right half showing the same scene rendered by MDWT at 1 spp. The rendering time for each slice is indicated on the left side of each image, and each image is accompanied by a line plot of the average rendering times in milliseconds (lower is better).

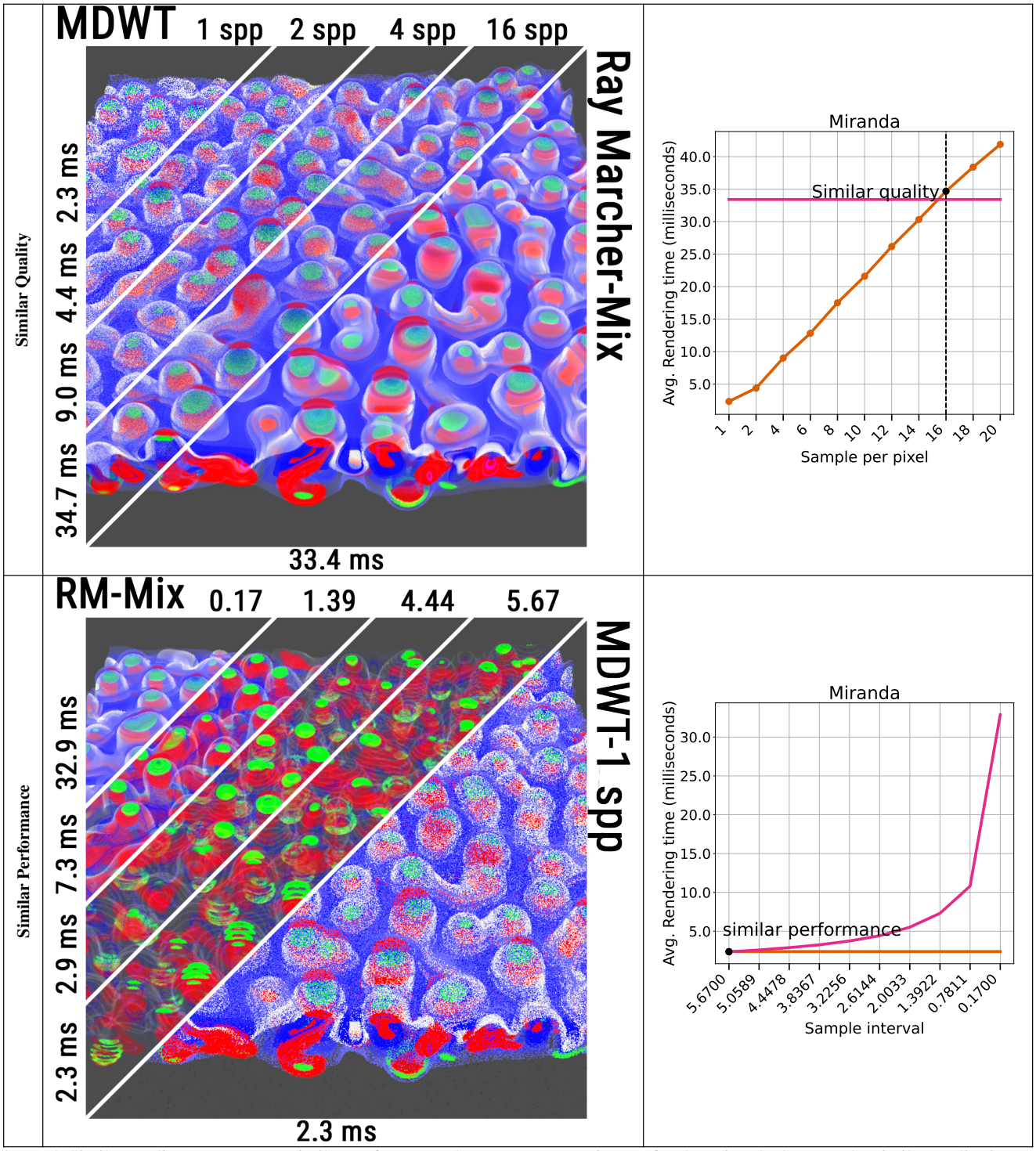


Figure 5: Similar quality (top row) vs. similar performance (bottom row) experiments for the Miranda dataset. The similar quality images display slices rendered with our Multi-Density Woodcock Tracking (MDWT) at increasing samples per pixel (spp), while the bottom-right half shows the ray marcher using mix blending with half of the minimum cell size as the sampling interval (Nyquist rate). The similar performance images show slices rendered with the ray marcher and mix blending (RM-Mix) for increasing sampling intervals, with the bottom-right half showing the same scene rendered by MDWT at 1 spp. The rendering time for each slice is indicated on the left side of each image, and each image is accompanied by a line plot of the average rendering times in milliseconds (lower is better).

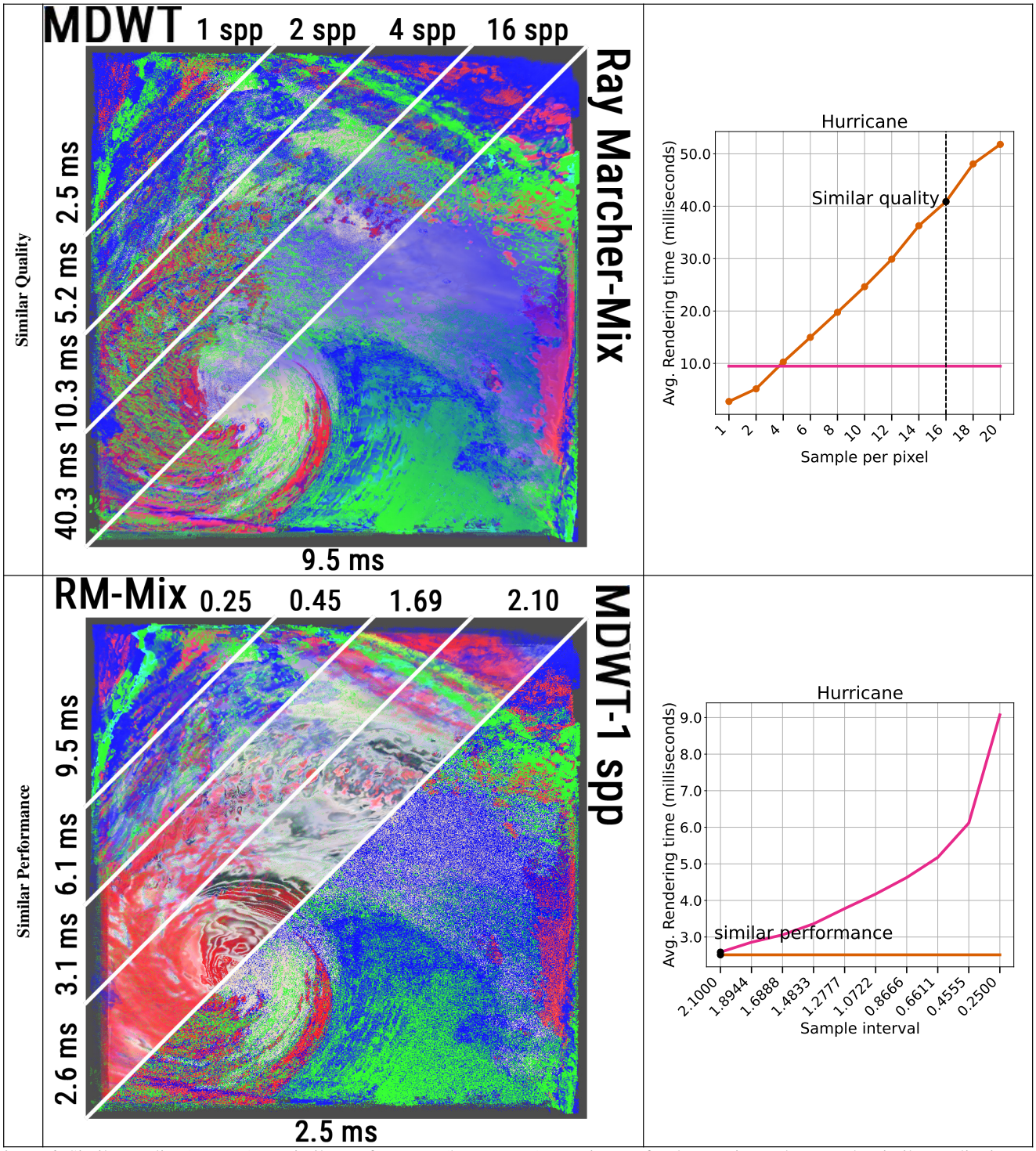


Figure 6: Similar quality (top row) vs. similar performance (bottom row) experiments for the Hurricane dataset. The similar quality images display slices rendered with our Multi-Density Woodcock Tracking (MDWT) at increasing samples per pixel (spp), while the bottom-right half shows the ray marcher using mix blending with half of the minimum cell size as the sampling interval (Nyquist rate). The similar performance images show slices rendered with the ray marcher and mix blending (RM-Mix) for increasing sampling intervals, with the bottom-right half showing the same scene rendered by MDWT at 1 spp. The rendering time for each slice is indicated on the left side of each image, and each image is accompanied by a line plot of the average rendering times in milliseconds (lower is better).