Monte Carlo Methods for 2D Flow Visualization Supplemental Material

Xingze Tian and Tobias Günther

Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

1. Results

We test both our approaches on slices of different vector fields. The BENZENE data set contains the three-dimensional electrostatic field around a benzene molecule. The BORRO and TREFOIL data set are three-dimensional simulations of magnetic fields in different ring and knot configurations [CB11]. Lastly, the CYLINDER data set contains a von-Kármán vortex street in a flow with Reynolds number 160 around a cylinder [Pop04,GGT17]. Fig. 1 compares evenly-spaced streamlines [JL97], line integral convolution [CL93], and our two proposed approaches on all four data sets. A high density of evenly-spaced streamlines may result in many suddenly ending lines. To provide a fairer comparison, we added a tapering to the streamlines (first column in Fig. 1), which was done in three steps. First, evenly-spaced streamlines were computed with the classic algorithm of Jobard and Lefer [JL97], during which the seed points were stored. Second, the lines were discarded and replaced with full streamlines computed from the seed points, resulting in lines that end when they reach a critical point or exit the domain. And third, all lines were rendered with super-sampling in random order, giving each line a finite width (black) and a thin halo (white). By starting from the seed points of the evenly-spaced streamlines, it is guaranteed that the domain is sufficiently densely sampled. The line integral convolution (second column) was contrast-enhanced for all test scenes to serve as baseline. Our smooth vector graphics formulation (third column) visually separates larger flow structures well due to the higher contrast and the distinct color gradients. The light transport simulation (fourth column) exhibits a darkening effect where lines end or when the line density decreases due to diverging flow. Unlike all other approaches, it creates a depth impression that makes the scene more haptic. In the following, we measure the performance of the two approaches.

2. Performance Measurements

We measured the computation time for both our approaches on an Intel i9-10980XE CPU (3.0 GHz) with an NVIDIA RTX 1080 GPU. All measurements are taken for an image resolution of 512×512 pixels. The smooth vector graphics approach is implemented on the GPU, while we targeted Mitsuba to the CPU for the light transport simulation. Table 1 lists the parameters of all test scenes, as well as the computation times. The numerical integration step size of the RK4 integrator is determined automatically by the Courant-Friedrich-Levy condition for a Courant number of $C = 0.5$, meaning that the integration step is bounded such that the largest possible integration step cannot move further than 0.5 grid cells. Both our approaches take a set of evenly-spaced streamlines [JL97] as input. We report the initial separation distance d_{sep} and the testing distance d_{test} for both approaches, and list the resulting number of streamlines N that are rendered per data set. The fading distance F, as well as the distances d_{sep} and d_{test} are specified relative to the size of a pixel Δ. Let $X \times Y$ be the image resolution, and let (x_{min}, y_{min}), and (x_{max}, y_{max}) be the corners of the domain bounding box, then the pixel size Δ is:

$$\Delta = \min(\Delta_x, \Delta_y), \quad \Delta_x = \frac{x_{max} - x_{min}}{X-1}, \quad \Delta_y = \frac{y_{max} - y_{min}}{Y-1} \quad (1)$$

Thus, $d_{sep} = 10$ corresponds to a separation distance of 10Δ. If not mentioned otherwise, all SVG images are rendered with 100 samples per pixel (spp), and all LTS images are rendered with 1,024 samples per pixel (spp). The smooth vector graphics rendering takes 116-275 seconds for 100 spp, which averages at about 1.2-2.8 seconds per iteration. The time for the light transport simulation remained consistently at about 75 seconds for 1,024 spp, which corresponds to about 73 milliseconds per iteration.

3. Convergence Sequence

Since both our approaches are computed via Monte Carlo integration, each image needs a number of iterations to converge. In the previous section, we have seen that the light transport solver needs orders of magnitude less time per iteration. However, both approaches might converge at different rates. Does the smooth vector graphics renderer perhaps need fewer iterations for a decent image? In Fig. 2, we depict a convergence sequence for both approaches, in which the results of early iterations are depicted. It turns out that the smooth vector graphics renderer indeed converges with much fewer iterations compared to the light transport simulation. With both approaches, useful previews are obtained after a few seconds, which allows for potential parameter adjustment.

4. Post-Processing for Contrast Enhancement

Prior state-of-the-art work on LIC, such as FastLIC [HS98], applies a post-processing to enhance the contrast of LIC visualiza-

Figure 1: *Results of evenly-spaced streamlines [JL97], line integral convolution [CL93], and our two approaches in the four data sets.*

tions. In Figs. 3–6, we apply three contrast enhancement methods, which are all implemented in Matlab. The first, called `imadjust`, scales the data values, such that 1% of the lower and upper range of the values is saturated at low and high intensities, respectively. The second method, named `histeq`, applies a standard histogram equalization. The third method, called `adapthisteq`, performs histogram equalization locally in small regions rather than on the full image at once. Table 2 reports quantitative metrics for the difference between the image without contrast enhancement and the image with contrast enhancement, for all data sets and all four visualization methods.

For the evenly-spaced streamlines, `histeq` turns out to be counter-productive as it darkens the image, while `imadjust` and `adapthisteq` have a negligible effect. This can also be seen in the metrics. While `imadjust` has no effect at all, `adapthisteq` has almost no difference, and `histeq` shows a quantitative difference in all metrics, i.e., in SSIM, RMSE, and PSNR.

		Smooth Vector Graphics (SVG)						Light Transport Simulation (LTS)					
Dataset	Figure	d_{sep}	d_{test}	N	F	spp	time (sec)	d_{sep}	d_{test}	N	F	spp	time (sec)
BENZENE	Fig. 1	10	1.8	3,614	50	100	202.57	2	0.9	12,848	20	1,024	73.31
BORRO1	Fig. 1 (paper)	10	1.8	1,769	50	100	273.44	2	0.9	8,748	40	1,024	74.45
BORRO2	Fig. 1	10	1.8	1,677	50	100	275.80	2	0.9	8,296	40	1,024	73.56
BORRO3	Fig. 2	10	1.8	1,818	50	100	273.50	2	0.9	8,702	40	1,024	75.00
TREFOIL	Fig. 1	10	2.25	1,604	50	100	206.61	2	0.9	9,193	40	1,024	73.40
CYLINDER	Fig. 1	40	2.7	664	100	100	116.25	2	0.9	7,741	80	1,024	74.14

Table 1: *Performance measurements and parameters for all test scenes. Input to our approaches is a set of N evenly-spaced streamlines, computed by the method of Jobard and Lefer [JL97], using the separation distance d_{sep} and the testing distance d_{test}. Both distances as well as the fading distance F are measured in pixels. The time (in seconds) is reported for the total number of samples per pixel (spp).*

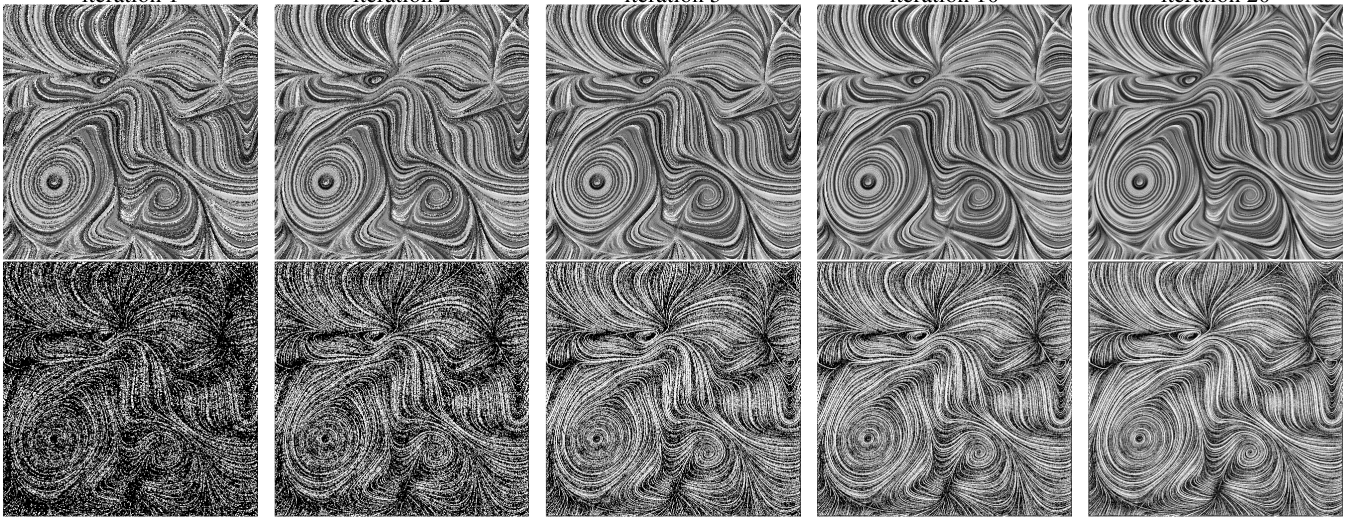

Figure 2: *A convergence sequence for both proposed Monte Carlo methods on* BORRO3*, showing how the noise reduces quickly after few iterations. The top row shows the smooth vector graphics (SVG) approach, and the bottom row shows the light transport simulation (LTS).*

Visually, we observe that `imadjust` was most effective in the LIC visualizations, which is why we applied this method in the LIC images throughout the paper. This is also the approach used in an open source FastLIC implementation [BS05]. From the metrics it can be seen that the post-processing had a significant effect. The `adapthisteq` enhancement caused a similar amount of change in the BENZENE and the CYLINDER data set for LIC as it did for our smooth vector graphics approach.

For our smooth vector graphics approach the post-processing has a minor effect. Apart from the aforementioned case of `adapthisteq` being applied to BENZENE and CYLINDER, all metrics (SSIM, RMSE, and PSNR) have been significantly better for the smooth vector graphics approach than for LIC.

With our light transport simulation approach, the `histeq` is counter-productive as it darkened the image too much. Both `imadjust` and `adapthisteq` have again a negligible effect. Quantitatively, the contrast enhancement had less effect on the light transport simulation method than on the smooth vector graphics method. From these experiments, we conclude that post-processing is not strictly necessary with our approaches. As was known from prior work, post-processing is essential for LIC.

References

[BS05] BIGDELY-SHAMLO N.: Vfv toolbox, 2005. https://github.com/bigdelys/vector_field_vis. 3

[CB11] CANDELARESI S., BRANDENBURG A.: Decay of helical and nonhelical magnetic knots. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics 84*, 1 (2011), 016406. 1

[CL93] CABRAL B., LEEDOM L. C.: Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1993), SIGGRAPH '93, Association for Computing Machinery, p. 263–270. doi:10.1145/166117.166151. 1, 2, 5

[GGT17] GÜNTHER T., GROSS M., THEISEL H.: Generic objective vortices for flow visualization. *ACM Transactions on Graphics (Proc. SIGGRAPH) 36*, 4 (2017), 141:1–141:11. 1

[HS98] HEGE H.-C., STALLING D.: Fast LIC with piecewise polynomial filter kernels. *Mathematical Visualization: Algorithms, Applications and Numerics* (1998), 295–314. doi:10.1007/978-3-662-03567-2_22. 1

[JL97] JOBARD B., LEFER W.: Creating evenly-spaced streamlines of arbitrary density. In *Visualization in Scientific Computing '97* (Vienna, 1997), Lefer W., Grave M., (Eds.), Springer Vienna, pp. 43–55. 1, 2, 3, 4

[Pop04] POPINET S.: Free computational fluid dynamics. *ClusterWorld 2*, 6 (2004). URL: http://gfs.sf.net/. 1

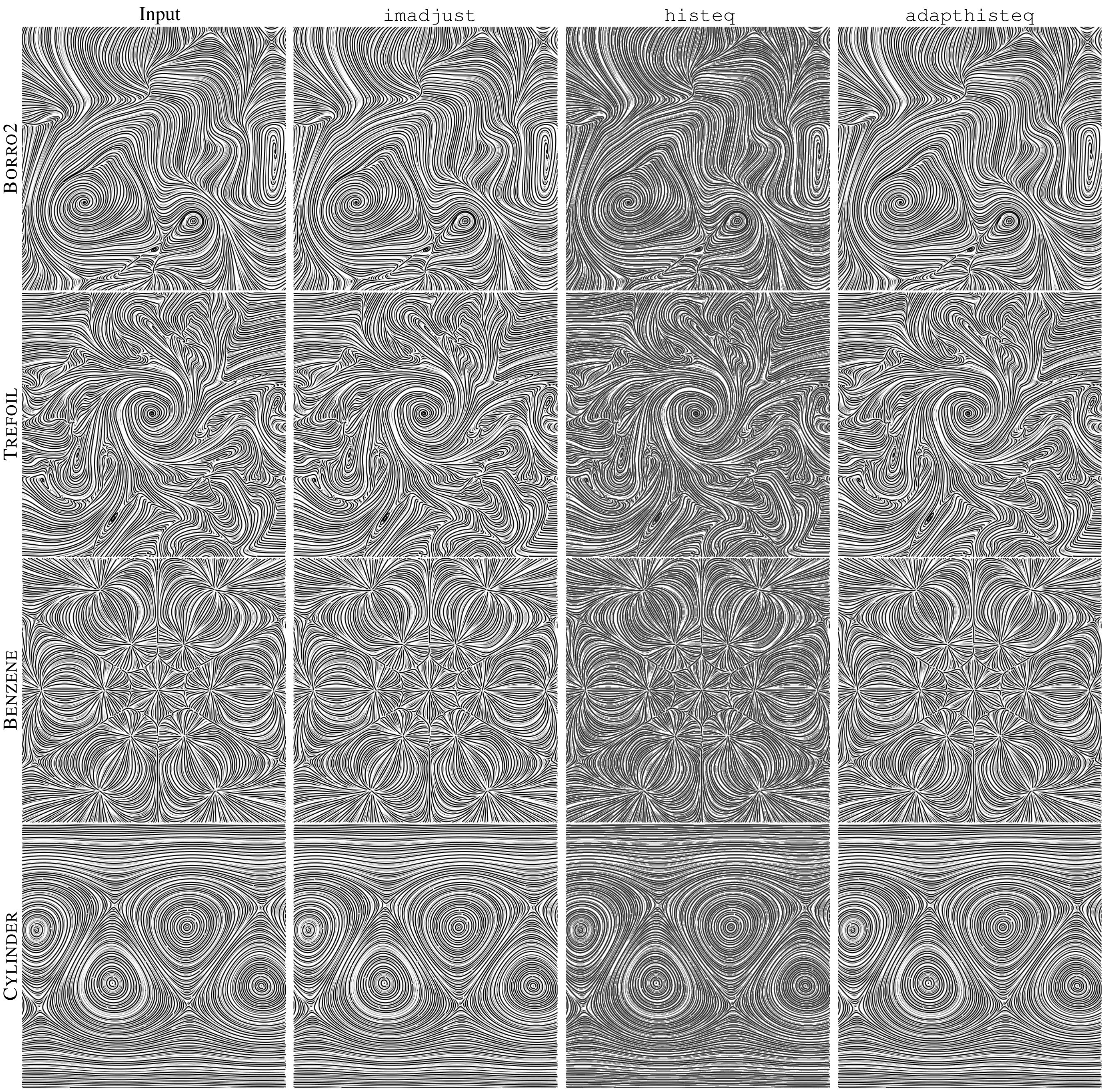

Figure 3: *Application of contrast enhancement methods for the post-processing of evenly-spaced streamlines [JL97].*

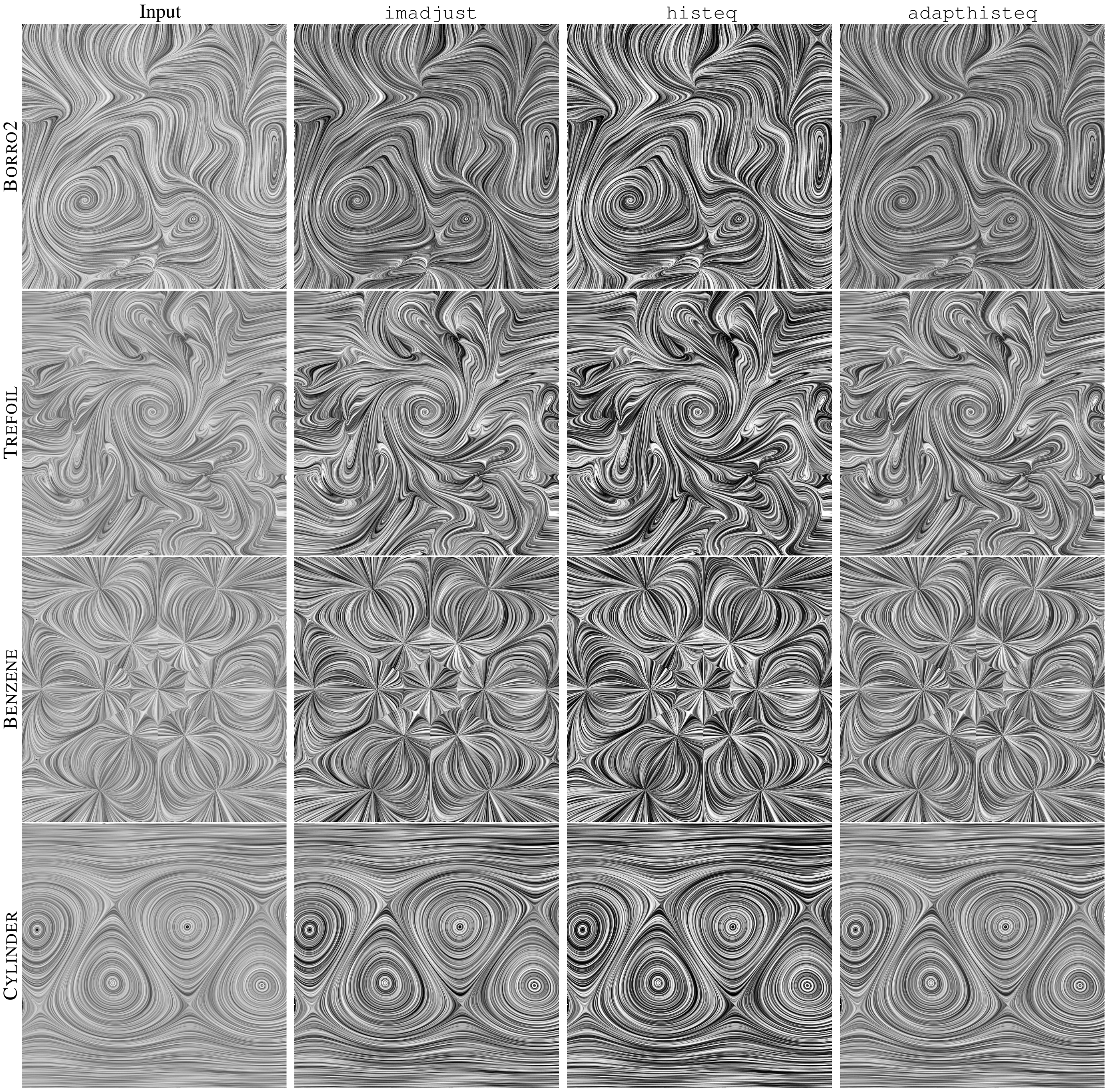

Figure 4: *Application of contrast enhancement methods for the post-processing of line integral convolutions [CL93].*

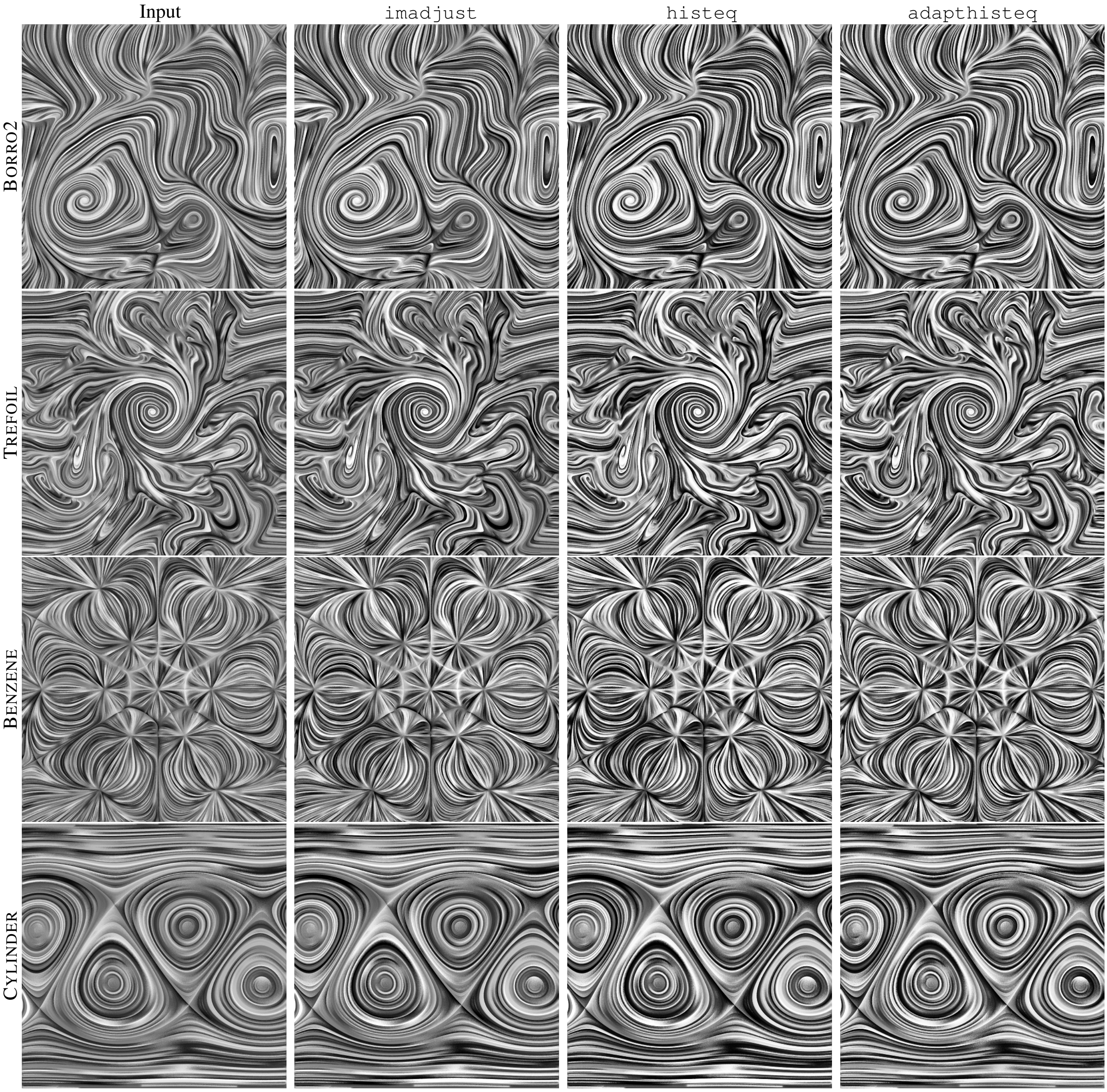

Figure 5: *Application of contrast enhancement methods for the post-processing of our smooth vector graphics approach.*

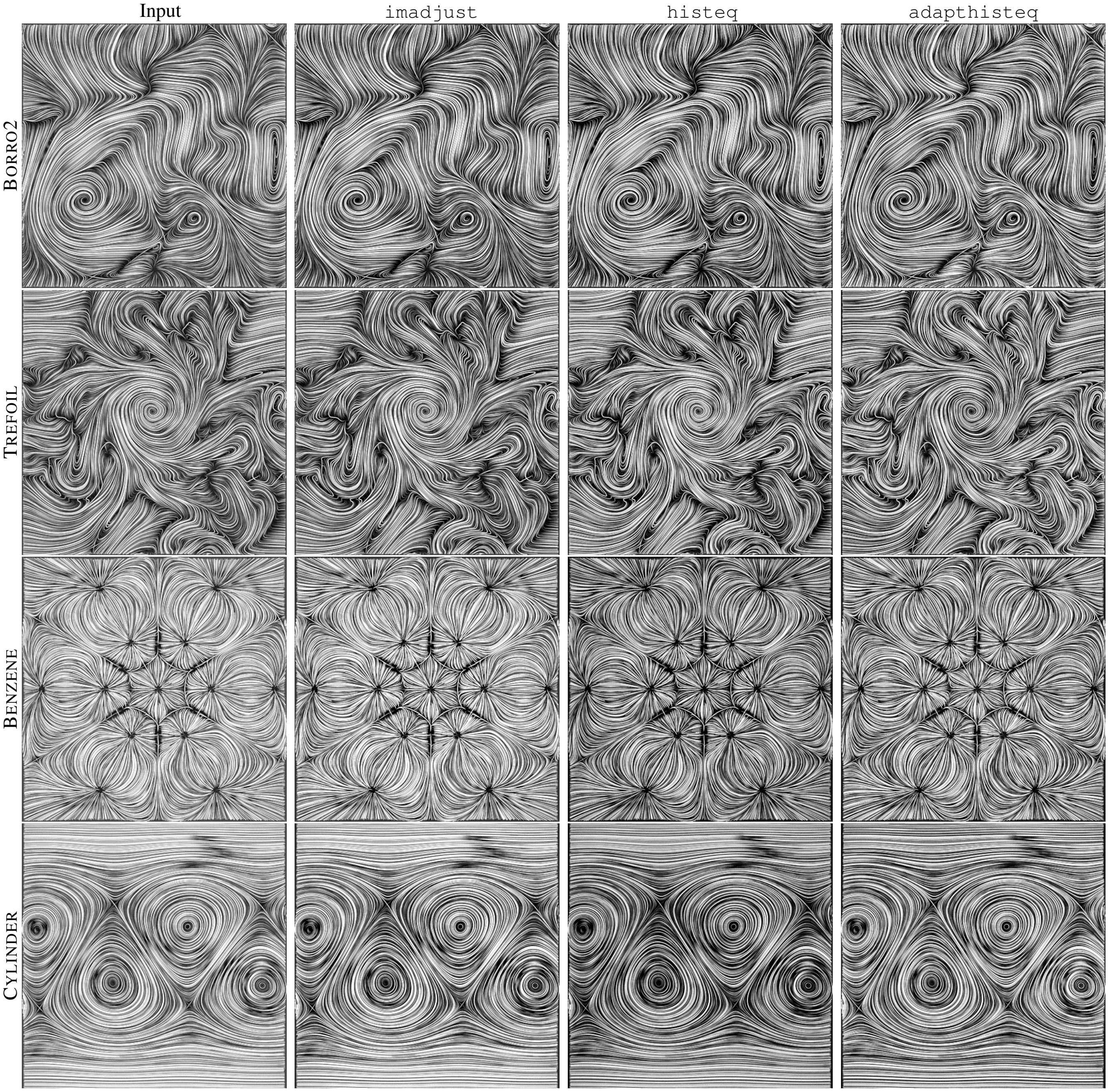

Figure 6: *Application of contrast enhancement methods for the post-processing of our light transport simulation approach.*

Data set	Method	Evenly-spaced			Line integral convolution			SVG (ours)			LTS (ours)		
		SSIM↑	RMSE↓	PSNR↑	SSIM↑	RMSE↓	PSNR↑	SSIM↑	RMSE↓	PSNR↑	SSIM↑	RMSE↓	PSNR↑
BORRO2	`imadjust`	1.0000	0.0000	∞	0.4946	39.2263	16.2593	0.9813	9.4662	28.6073	0.9869	11.3712	27.0147
	`histeq`	0.8947	41.9222	15.6819	0.3457	60.4029	12.5097	0.8983	24.0656	20.5029	0.9648	16.8957	23.5753
	`adapthisteq`	0.9969	7.4426	30.6963	0.5435	33.0318	17.7522	0.8936	24.8920	20.2096	0.9613	18.8085	22.6437
TREFOIL	`imadjust`	1.0000	0.0000	∞	0.8406	32.2618	17.9570	0.9812	9.7242	28.3737	0.9921	7.7739	30.3180
	`histeq`	0.9017	40.4111	16.0008	0.6733	54.8102	13.3536	0.9068	22.7796	20.9799	0.9712	13.5449	25.4953
	`adapthisteq`	0.9971	7.3935	30.7538	0.8749	29.0006	18.8827	0.9013	24.2755	20.4275	0.9666	17.4583	23.2908
BENZENE	`imadjust`	1.0000	0.0000	∞	0.8402	34.8589	17.2845	0.9675	12.9666	25.8742	0.9860	11.6198	26.8268
	`histeq`	0.8992	41.0343	15.8679	0.6850	54.7550	13.3623	0.8742	27.3019	19.4070	0.9235	39.4217	16.2161
	`adapthisteq`	0.9966	8.0024	30.0664	0.8740	30.0843	18.5640	0.8709	28.0216	19.1810	0.9444	30.5869	18.4201
CYLINDER	`imadjust`	1.0000	0.0000	∞	0.8142	34.6593	17.3344	0.9719	11.0461	27.2667	0.9787	14.8935	24.6708
	`histeq`	0.9079	38.8724	16.3380	0.6380	56.8151	13.0415	0.8853	24.6589	20.2913	0.9181	38.9723	16.3157
	`adapthisteq`	0.9967	7.7819	30.3091	0.8725	27.0444	19.4892	0.8705	26.3995	19.6989	0.9313	32.1677	17.9824

Table 2: *This table reports the differences between visualizations with and without contrast enhancement, here listed for all data sets and for three different contrast enhancement methods. For comparison, SSIM (higher is better), RMSE (lower is better), and PSNR (higher is better) are reported. PSNR is measured in dB.*