



# Optimization of Opacity and Color for Dense Line Sets Additional Material

Berkan Tuncay  and Tobias Günther 

Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

## 1. Single vs. Complete Linkage

The distance matrix  $\mathbf{D} \in \mathbb{R}^{m \times m}$  contains the distance between each pair of lines, where  $m$  corresponds to the total number of lines. Initially, each line is considered to be its own cluster. The most similar clusters are identified by locating the minimum element in the distance matrix. When two clusters  $i$  and  $j$  merge, the corresponding rows and columns in the distance matrix are updated. A commonly used update rule is the Lance-Williams dissimilarity update formula [MC12].

$$d(i \cup j, k) = \alpha_i d(i, k) + \alpha_j d(j, k) + \beta d(i, j) + \gamma |d(i, k) - d(j, k)| \quad (1)$$

**Complete Linkage.** The update formula can describe a wide range of linkage methods such as the complete linkage criterion. It is defined with  $\alpha_i = \alpha_j = \gamma = \frac{1}{2}, \beta = 0$  and the expression simplifies to the maximum distance between all possible pairs of elements in each group.

$$d(i \cup j, k) = \frac{d(i, k) + d(j, k) + |d(i, k) - d(j, k)|}{2} \quad (2)$$

$$= \max(d(i, k), d(j, k)) \quad (3)$$

The criterion produces highly compact clusters, as it requires all intra-cluster distances to be small.

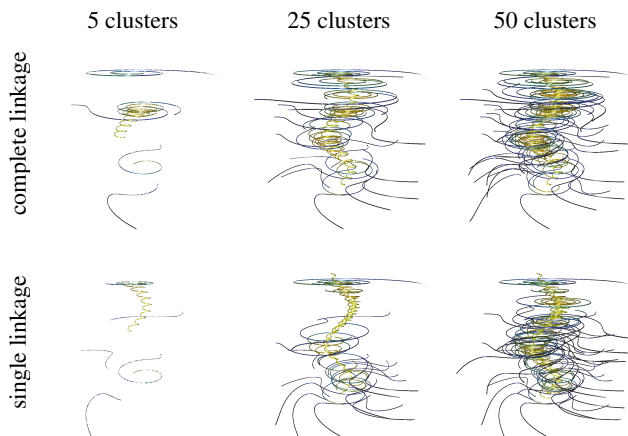
**Single Linkage.** Another commonly used method is the single linkage criterion. It is defined as the minimum distance over all possible pairs of each cluster. In terms of the Lance-Williams dissimilarity update formula it sets  $\alpha_i = \alpha_j = \frac{1}{2}, \gamma = -\frac{1}{2}$  and  $\beta = 0$ .

$$d(i \cup j, k) = \frac{d(i, k) + d(j, k) - |d(i, k) - d(j, k)|}{2} \quad (4)$$

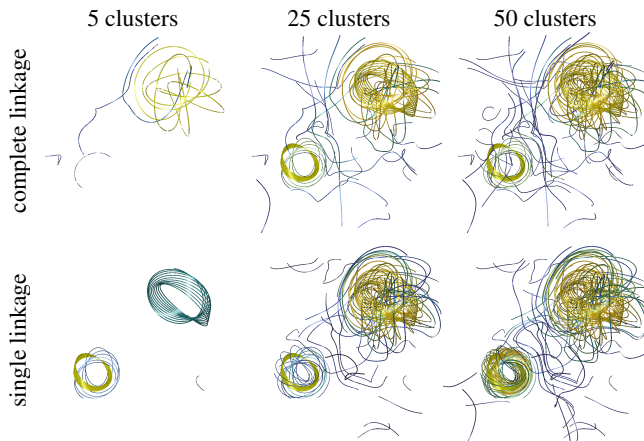
$$= \min(d(i, k), d(j, k)) \quad (5)$$

Single linkage produces more chain-like clusters since the nearest neighbors between different clusters are merged. This may be misleading in cases where two clusters are vastly different with the exception of a few singular points. With single linkage, such clusters may be erroneously combined.

In Fig. 1 and Fig. 2, we compare the different linking methods for the TORNADO and BORROMEAN datasets. Opacity optimization is disabled, to clearly see the lines which are selected. In Fig. 1, the



**Figure 1:** Visualization of the TORNADO dataset using complete and single linkage for different numbers of clusters.



**Figure 2:** Visualization of the BORROMEAN dataset using complete and single linkage for different numbers of clusters.

single linkage criterion seems to favor clusters around the vortex core line of the tornado. Other long outreaching chains of lines are also clustered and chosen as representatives. The complete link-



**Figure 3:** Methods for representative selection. The complete BORROMEAN dataset was reduced to a single cluster, which represents one of the rings. The leftmost image showcases the mean representative, which generates a curve that was not present in the data. The middle image illustrates the least distance representative, which displays the line closest to all other lines within a cluster. The last image shows the most important representative, which is the line with the maximum total importance inside a cluster.

age criterion produces tighter groups, since lines are clustered with the maximum intra-cluster distance. Therefore, the body of curved lines of the tornado is better represented. In Fig. 2, the single linkage criterion produces clusters around the long winding rings of the BORROMEAN dataset, whereas the complete linkage criterion struggles to capture this intrinsic structure. However, as the amount of clusters is increased, both criteria begin to approach each other. Empirically, we found that complete linkage delivered good results in all our data sets and was therefore chosen as default choice. If the fluid flow contains important structures resembling long chains of lines, then the single linkage criterion yields more fitting clusters. More suitable clusters enable us to reduce the total number of clusters, without losing relevant information.

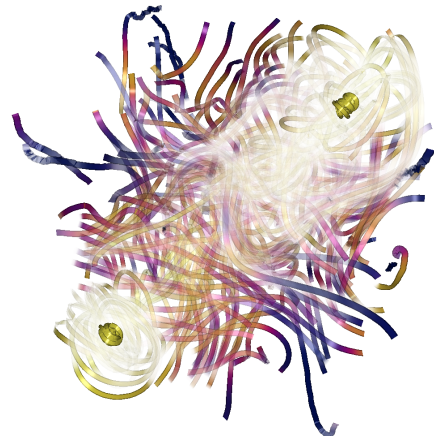
## 2. Selection of Representative Line

With the agglomerative clustering algorithm, we obtained a set of clusters, each cluster containing a varying number of lines, and the objective is to visualize each cluster by finding a representative line. A line representative may be selected from the already existing set of lines within a cluster or it may be constructed completely anew.

**Mean Line Representative.** In an attempt to capture all the information included in a cluster, the *mean line* representative is obtained by calculating the sum of all lines within a cluster followed by the division of the total number of lines. Unfortunately, this approach produces malformed lines due to difficulties arising from lines with different lengths and varying discretizations as illustrated in Fig. 3 (left). Furthermore, the newly constructed line is not guaranteed to be a streamline, making this an undesirable option.

**Least Distance Representative.** It is desirable to select representative lines that are contained in the clusters themselves. The *least distance* representative is the line with the smallest total distance to all other lines within its cluster. It is found by computing the filtered distance matrix  $\mathbf{D}_{\mathcal{C}}$ , which only consists of the rows and columns representing lines contained in the cluster  $\mathcal{C} = \{\bar{\mathbf{x}}_1, \dots, \bar{\mathbf{x}}_m\}$  and subsequently finding the minimum column sum.

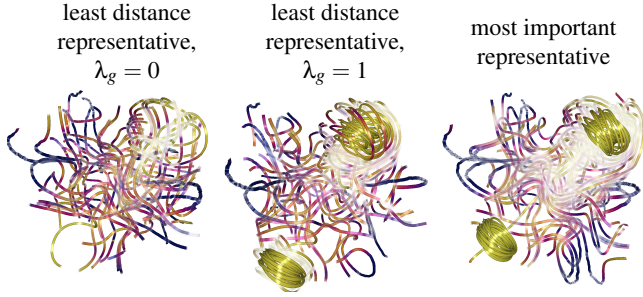
**Most Important Representative.** The aim of the last method is to be more in line with the objective of the opacity optimization.



**Figure 4:** Visualization of the BORROMEAN dataset using the mean line representative, which produces ill-formed lines. In this example, the rings collapsed, making this approach unsuitable.

Similarly to how important lines receive higher opacity values, important lines should also be preferably picked as cluster representatives. The most important representative is calculated by selecting the line with the highest average importance value along the line.

**Comparison.** In this section, we compare the three methods with each other and provide an overview of their advantages and disadvantages. Initially, the *mean line* representative was implemented, with the idea of condensing all lines of a cluster into a single one. Unfortunately, this approach creates malformed lines. Especially, intricate structures such as the rings of the BORROMEAN data set are completely mangled by the mean representative as shown in Fig. 4. Therefore, we introduced the *least distance* representative and the *most important* representative. Both methods select an existing line inside of the cluster instead of creating a new one. However, which line is deemed as most suitable differs in both methods. Their differences are investigated through the analysis of the cluster representatives they produce in Fig. 5. The *least distance* representative method picks the line with the lowest distance to all others. When the line distances only consist of the spatial dimension, then important structures may not be chosen as representatives. Therefore it may be necessary to incorporate the importance dimension in the line distance computations in order to ensure important representatives. The most important representative directly calculates the total importance of each line within a cluster and returns the one with the highest value. This approach guarantees the choice of important lines independent of the inclusion of importance in the distance metric. To summarize, the least distant representative should be used, when the objective is to get the most accurate depiction of the contents of each cluster. However, if we are more interested in the structures marked as important, then it is necessary to tune the importance scalar factor  $\lambda_g$  for the distance metrics. This process can also be completely sidestepped by applying the most important representative method. This method always returns the line with highest total importance per cluster irrespective of the intra-cluster distance values.



**Figure 5:** Visualization of the BORROMEAN dataset with different representative selection methods. The least distance representative method picks the line with the lowest distance to all others. When the line distance metric only considers the spatial dimensions, then the rings are not well represented. If the line distance metric incorporates the importance dimension, then the rings are clearly visible. The most important representative directly calculates the total importance of all lines within a cluster. Therefore, structures marked as important are included by default, independent of the usage of importance in the distance metric.

### 3. Effect of Bin Size on Performance

In this section, we investigate the impact of the bin size  $N$  on the total computation time and on the histogram related stages. The test setup is identical to the performance measurement setup in the main paper. Here, we focus on the BORROMEAN dataset. The results are detailed in Table 1. The following trends emerge as the number

Bins	Histogram	CDF	Exp. decay	Total time
4	26.857	0.153	0.004	58.181
8	26.278	0.190	0.005	58.004
16	24.670	0.206	0.005	56.179
32	17.688	0.183	0.005	48.993
64	16.478	0.127	0.005	47.997
128	15.332	0.153	0.005	46.501
256	12.775	0.110	0.005	44.004
512	12.020	0.091	0.005	43.726
1024	10.330	0.072	0.006	41.782

**Table 1:** Computation times of the histogram-related stages for the BORROMEAN dataset in milliseconds.

of bins is increased. The time to calculate the histogram and its CDF decreases, whereas the time to compute exponential decay largely remains unaffected. To calculate the histogram, the entire fragment linked list buffer is iterated and each coloration and visibility value is read. The coloration value is then used to determine in which bin or, more precisely, in which pixel of the histogram texture the visibility value is written. The additive blending into the same pixel of the render target leads to overdraw, which causes the slowdown. Increasing the number of bins generally leads to a more even distribution of coloration values within the bins which in turn leads to less overdraw. Consequently, the performance improves when the number of bins is increased. The time required to compute the CDF of the histogram also decreases as the number of bins is increased. The CDF is calculated using the hills and steele

scan [HS86]. With low bin counts, the algorithm performs worse than the serial implementation, due to its overhead. However, as the number of bins increases, and with it the number of threads, the computation time reduces. Lastly, exponential decay is almost completely unaffected by the number of bins. The time required to compute the exponentially damped CDF only slightly changes with varying bin sizes. Theoretically, this problem scales linearly with bin size, however due to its embarrassingly parallelizable nature it is practically solved in near-constant time.

### 4. Effect of Cluster Size on Performance

In this section, we measure the effect of the cluster size on the individual rendering stages using an AMD Ryzen 9 5900X and an NVIDIA GeForce RTX 3070 TI. For this purpose, the CTBL dataset has been rendered with different cluster sizes for the otherwise identical performance measurement settings as described in the main paper, with the exception of line width. The line width was slightly reduced, since the original size leads to fragment linked list overflows at cluster size 1200. The results are detailed in Table 2. It is evident that the fragment linked list sorting stages are

Stage	400	600	800	1000	1200
Create list (LR)	0.75	1.24	1.61	1.97	2.27
Sort list (LR)	63.59	87.77	105.16	138.00	157.79
Comp. alpha	1.84	1.90	2.47	2.38	1.97
Smoothing	0.03	0.03	0.03	0.03	0.03
Fade alpha	0.04	1.25	0.07	0.08	0.09
Create list	0.89	1.07	1.60	1.91	2.19
Sort list	65.04	79.89	90.02	87.72	114.50
Histog.	2.19	2.67	2.73	2.87	2.32
Histog. CDF	0.01	0.26	0.01	0.01	0.01
Exp. decay	0.01	0.01	0.01	0.01	0.01
Render image	0.68	0.82	0.80	0.86	1.01
Total time	137.50	177.25	206.69	238.64	284.00

**Table 2:** Computation times in milliseconds per rendering stage for a different number of clusters  $N$  in the CTBL dataset.

affected the most by the increase of the cluster size. This is because the number of clusters (and thereby representative lines) increases, which results in longer fragment linked lists. This is the case in the CTBL dataset, which was visualized in the main paper in Fig. 5 (bottom right). The streamlines densely cover the viewport and have a high degree of overlap. The extent to which lines overlap is further increased due to the long length of most lines. Therefore, each pixel will contain a number of lines rasterized into it, which produces long fragment linked lists. While not comparable to the sorting stages, the draw time of the list creation stages and alpha computation stage also increase with the cluster count.

### References

- [HS86] HILLIS W. D., STEELE G. L.: Data parallel algorithms. *Commun. ACM* 29, 12 (dec 1986), 1170–1183. 3
- [MC12] MURTAGH F., CONTRERAS P.: Algorithms for hierarchical clustering: an overview. *WIREs Data Mining and Knowledge Discovery* 2, 1 (2012), 86–97. doi:<https://doi.org/10.1002/widm.53>. 1