Workshop on Visualisation in Environmental Sciences (EnvirVis) (2021)
S. Dutta and K. Feige and K. Rink and D. Zeckzer (Editors)

# GPU-Assisted Visual Analysis of Flood Ensemble Interaction

D. W. Johnson[1†] and T. J. Jankun-Kelly[2]

[1]National Water Center, NOAA, United States
[2]Department of Computer Science & Engineering, Mississippi State University, United States

**Abstract**

*Analysis of overlapping spatial data sets is a challenging problem with tension between clearly identifying individual surfaces and exploring significant overlaps/conflicts. One area where this problem occurs is when dealing multiple flood scenes that occur in an area of interest. In order to allow easier analysis of scenes with multiple overlapping data layers, we introduce a visualization system designed to aid in the analysis of such scenes. It allows the user to both see where different data sets agree, and categorize areas of disagreement based on participating surfaces in each area. The results are stable with regard to render order and GPU acceleration via OpenCL allows interaction with large datasets with preprocessing dynamically. This interactivity is further enhanced by data streaming which allows datasets too large to be loaded directly onto the GPU to be processed. After demonstrating our approach on a diverse set of ensemble datasets, we provide feedback from expert users.*

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications—Visualization H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces

## 1. Introduction

Flooding is one of the most frequently occurring natural disasters; flood modeling is one tool used to mitigate their effects. These models can be used to predict how flood control projects would influence both historical and future floods; however, as the number of possible mitigation scenarios increases, the number of modeled outputs increases exponentially. Moreover, in such disaster scenarios, disagreements or conflicts in the model ensembles are often as important as where they agree—areas where disarrangement occurs are the area's which require human decisions. Due to the multiplicity of model simulations, their integration and the identification of conflict is a challenge. To address these difficulties, we propose a scalable visualization system that is designed to allow rapid analysis of multiple data surfaces. Although the motivation for designing this system was for flood scene analysis, it can be applied to any dataset where there are multiple overlapping data fields, such as those produced by weather model ensembles or other ensemble simulations of physical events. Our system quickly allows exploration of agreement and conflict in the overlapping surface with interactive exploration of the combinatorial space of possible interactions. Using an overview+detail approach, we also facilitate exposing the diverse collection of overlaps for any number of interactions. By using the GPU for computation and visualization, the systems increases the amount of user flow [EVMJ*11] by reducing mental state switches or iterative search enforced.

Current approaches to analyzing multiple overlapping spatial scalar datasets are costly in time and money. One contributing factor to this cost is that GIS tools display surfaces with ordered painting, where first one surface is drawn then the next and so forth until all surfaces have been displayed. This causes surfaces that are displayed later to obscure surfaces that are earlier in the rendering sequence. Reordering surfaces and transparency are ways that a user can attempt to view obscured information, but such reordering becomes cognitively prohibitive as the number of surfaces/ensembles increase. Transparency is not a complete solution either as the results of displaying multiple transparent surfaces is dependent on the order of rendering. In addition, blending of too many surfaces prevents the user from being able to determine what surfaces were blended at any given point. When dealing with ensemble output, transparency can be effectively used to show areas of agreement but can not effectively identify sub-regions where there is partial agreement between different sets of ensemble members. All of these problems are made more severe by the fact that any given output can easily exceed multiple gigabytes for storage and rendering is not an instantaneous operation, particularly when done in software. These same problems also occur when trying to analyze multiple outputs describing different historical or statistical events.

To address these problems, we contribute a visualization system with the following attributes:

- Visual appearance is not influenced by the order in which inputs are processed.
- Areas where multiple surfaces are interacting are indicated.

† Work completed while at Mississippi State University



DOI: 10.2312/envirvis.20211077

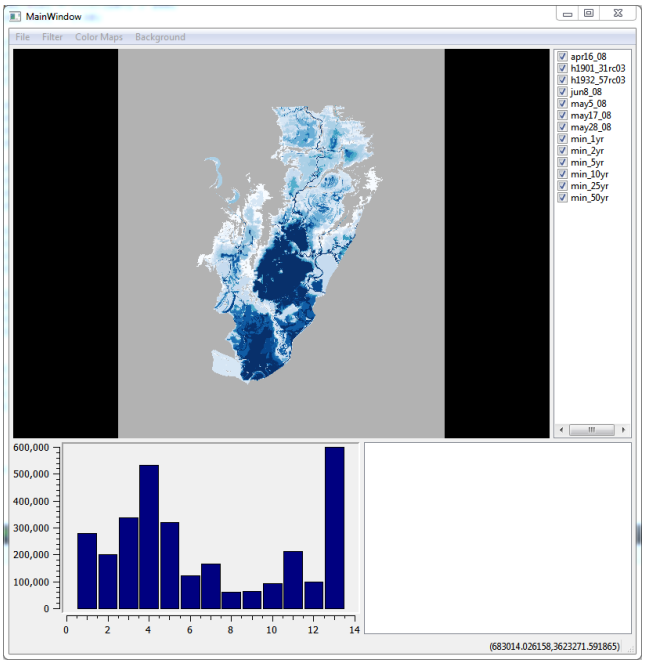



**Figure 1:** ***Overview*** *Summary view of per-pixel overlap from flood models of the Yazoo Backwater area. The spatial region (upper middle) uses a blue-white saturation ramp to indicate agreement (more saturation, more agreement); the histogram (lower left) summarizes the distribution of each level of agreement from no overlap at a pixel on the left to maximal overlap on the right. The two visual elements allow the level of overlap at any position to be determined.*

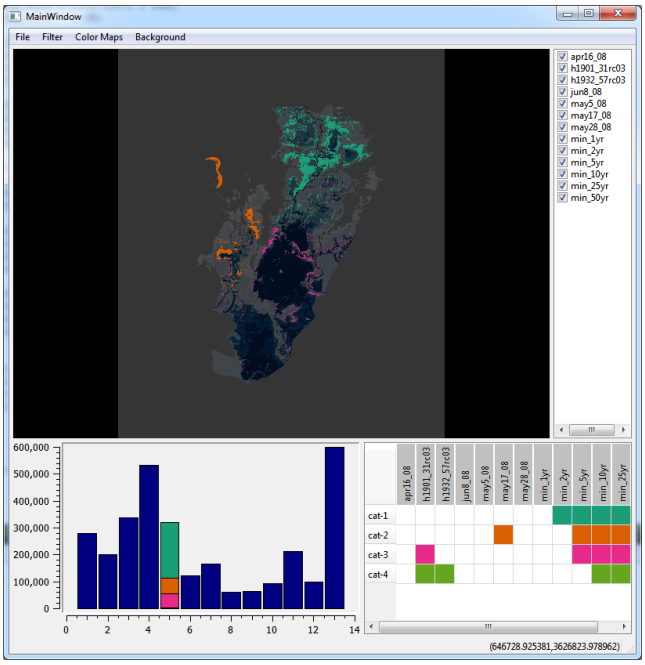



**Figure 2:** ***Detail*** *By selecting a histogram bin, interacting surfaces for that bin are highlighted; in this example, we examine locations where five surfaces overlap. Unique categorical colors per group are used both in the spatial display and the overlap matrix (lower right) to identify the involved surfaces. The histogram changes to show the percentage of pixels each group occupies. In this case, the overlapping surface with the most coverage consists of the min_2y, min_5y, min_10y, min_25y, and min_50y surfaces (shown in teal).*

- Both areas of high multi-surface agreement and conflict are visually indicated.
- Surfaces that compose regions of interaction are quickly identified.
- Data prepossessing is minimal.
- Works with massive datasets and or large numbers of simultaneous inputs via GPU-accelerated OpenCL kernels.
- Results can be imported into existing geo-spatial tools.

These features combine into a visualization approach that is better integrated into experts workflows, allowing more iterations of exploration. The details of our system, a set of case studies with diverse inputs, and an evaluation by experts are discussed in the following sections.

## 2. Related Work

Multivariate visualization is an active field of research in which many potential solutions have been proposed. In Taylor [Tay02], several methods for visualizing multiple fields on a single surface are summarized including texture based techniques, a spot based blending technique (Data Driven Spots or DDS [Bok03]), and a line segment representation technique (Oriented Slivers [WEL*00]). The DDS technique works by combining spots of distinct color and varying alpha/transparency values, with the alpha values based on the scaled value of the field being visualized. The color and alpha value resulting from the spot and data combination is then blended onto a background that is originally neutral gray. The spots used for each field have differing sizes, and animation is used to move the spots across the surface to better sample the data. The Oriented Slivers used small rectangles (slivers) with different orientations to represent each variable. The size of a sliver could be used to indicate the intensity at that position on the surface.

In Forsell et al. [FSL05], a technique that used three dimensional glyphs as well as color to visualize multiple variables on a single surface was tested. Soon after, Cai et al. [CDM06] presented a technique that used two layers to display information from multiple remotely sensed images. The first layer is an overview resulting from conversion of the multiple greyscale inputs into color; the second layer is composed of pie chart glyphs, where the filling of the pie chart is based on the relative strength of the input for each layer. The two views are then blended together with the relative strength of each layer depending on the size of the currently displayed area. Hsiao's dissertation [Hsi10] reviews several ways of displaying multiple fields of data on a single surface, including DDS and glyph based techniques, with examples given for both techniques. Cai et al. [CDM10] demonstrated the use of dual layer visualization, DDS, and Oriented Slivers for displaying remotely sensed hyperspectral imagery. In addition, visualizations were made that combined all three techniques.

Another area of relevant research is ensemble visualization, which deals with visualization of multiple model outputs where the input parameters for each model run are adjusted slightly. This area of research is important because some of targeted data groups in this paper utilize multiple flood scenes showing the effects of different flood control features; such data could be considered an ensemble dataset. Phadke et al. [PPA*12] showed how a sphere-based technique, a 3D extension of DDS, and a glyph based technique similar to the one shown in Hsiao [Hsi10] could be adapted to ensemble visualization. Specifically, ensemble visualizations were made by allowing agreement to modify the size, the opacity, or color of existing elements. Another approach to ensemble visualization is shown in Alabi et al. [AWH*12] where ensemble members are compared by slicing the modeled space of the simulations. The first slice is rendered with results from the first model, the second from the second model and so forth. This causes clear visual discontinuity where ensemble outputs differ. Other ensemble techniques summarize ensemble agreement and conflict via statistical measures and aggregation, such as the work in Noodles [SZD*10].

House et al. [BHW06] introduced a technique for viewing simultaneous three dimensional surfaces using textures to allow differentiation of the viewed surfaces. Guidelines for the generation of textures for this purpose were also given [HBW05]. In both, the scope was limited to two simultaneous surfaces.

The primary difference between our work and the previous is that previous attempts focused on the simultaneous visualization of the multiple data surfaces and the values of the depicted data on the surfaces. In contrast, the data inputs to our system are binary—they represent either existence (this location is/is not flooded) or threshold (this variable is above/below a critical value). Therefor, it is not necessary to display value—only presence. This allows more effort to be placed on the analysis of surface interaction, which is not directly addressed in previous work. The number of possible interactions is restricted to a manageable number by limiting analysis to a single level of overlap at a time after an overview of all interactions is presented. While CPU-only solutions are possible, the acceleration due to the GPU allows dynamic exploration without pre-processing all combinations—a combinatorial explosion of space unless an exploration path is assumed, which our approach does not require. This approach is presented next.

## 3. Technique Overview

The proposed visualization system uses an overview+detail approach: The overview highlights areas of commonality and conflict (Figure 1), whereas the detail display allows drill-down into specific levels of overlap (Figure 2). The overview visualization (Figure 1) is created by summing the number of surfaces that define a given pixel for all locations in all surfaces; this is then visualized using a color ramp where saturation increases as the number of participating surfaces for a pixel increases. Due to the initial context of flood visualization, a white to blue ramp (low to high) was chosen. To the right of the main display is a list of surfaces being displayed. This list allows particular surfaces to be added or removed from the visualization; reordering is not supported because the final image is not dependent on the order that surfaces are processed (the details of the GPU-accelerated rendering algorithm are given in Section 4). The blank section to the lower right is reserved for sub-region categorization and is not used in the overview mode. Finally, a histogram showing the counts for each level of overlap is depicted below and to the left of the spatial image; the left of the histogram shows how many pixels had no overlap (1 participating surface) while the right shows how many had maximal overlap. The histogram allows easy access to a statistical overview about the degrees of overlap currently being visualized; the distribution indicates if there was significant agreement, conflict, or some combination of the two.

While the overview mode depicts where agreement and conflict occur, it does not indicate which groups of interacting surfaces make up those areas; this is the purpose of the focus mode (Figure 2). In this mode, all non-selected levels are desaturated, and a categorical colormap is used to identify each separate set of overlapping ("interaction") surfaces—e.g., in Figure 2, three major groups of five interacting surfaces take up the majority of the interactions (the teal, orange, and pink regions). While the spatial view indicates the position of the overlapping surfaces, the histogram changes such that the bar for the selected level of overlap indicates the percent contribution of each participating group of surfaces. Mousing over any sub-group on the selected histogram bar will display the percent contribution of that group and highlight the corresponding surfaces in the surface list. In this example, the teal set of overlapping surfaces make up more than half of the overlaps at this level. To clearly indicate which surfaces are members of each set of overlapping surfaces, the bottom right of the interface displays a matrix color coded by those groups. By reading the surface names at the top of the matrix marked in green, the five members of the teal interaction can be discovered quickly (with some scrolling, as necessary in this example); surface membership in other sets can be checked in the same manner. By selecting different bars in the histogram and inspecting the overlap matrix, surface interactions at each level can be iteratively stepped through. In addition, both the overview and focus modes allow the view point to be panned and or zoomed, to better visualize interactions at any particular area. These interactions can then be exported as GeoTIFF files for analysis or dissemination in other tools. Section 5 demonstrates how these interactive displays can be used for analysis.

## 4. Implementation

One of the major challenges encountered when processing multiple flood inundation maps or ensembles is the size of each input is potentially large. This makes direct processing on a CPU a poor choice when the system is designed to be used in an interactive manner. Processing with multiple CPUs is an option, particularly with the current abundance of multi-core processors. However, barring the use of supercomputing clusters, the most potent processing solution available is GPU programing. Modern GPUs have hundreds to thousands of stream processors and allow multiprocessing to a much higher degree than using multiple CPU cores and/or vector instruction sets. In addition, though all possible combinations of interactions could be pre-computed, the amount of storage to do so grows exponentially, an unnecessary cost when the GPU allows interactive computation of any combination of interactions. To al-

low the system to work on the widest possible range of devices, OpenCL was chosen to communicate with the GPU.

There are several challenges to processing data on a GPU:

1. GPU memory is much more limited than CPU memory.
2. Data transfer to and from the GPU is limited by the bandwidth of the bus.
3. The linear format of CPU image data does not interact ideally with a GPU processing data in rectangular blocks.

To address these problems, the OpenCL kernels were designed to be iterative and spatially partitionable. The iterative nature of the kernels means that only data about one surface (and the intermediary buffers that hold the processing state) needs to be resident on the GPU concurrently. Multiple surfaces are processed by streaming surface data from main memory to the GPU; the CPU is responsible for for resolving projection and registration of geographic data s the GPU deals only with image data. The kernels being spatially partitionable means that the results at any location depend only on the values defined at that location for all processed surfaces. As a result, when a single input surface, the necessary support buffers, and images can not be loaded into GPU memory due to the size of the input surface, processing can still ensue by segmenting the input into parts that are then processed sequentially.

Our systems uses separate algorithms for the two views (overview and focus); these make use of several OpenCL kernels (some shared) for processing. Section 4.1 explains the algorithms, while Section 4.2 shows how we achieved interactive performance. More in-depth details are available elsewhere [JJK14].

### 4.1. Algorithms

For the overview and focus algorithms, a series of accumulation buffers are used to count interactions and determine the sets of surfaces that compose each interaction level. This accumulation occurs over the surfaces in a way to ensure interactivity during large data processing. For the overview, the accumulation counts how many times a surface (e.g., flood event, ensemble member) occurs at that location over the datasets; for the focus, we accumulate participating surfaces. Data surfaces are loaded as GPU images, and another image is used for output.

The overview algorithm is run whenever the active set of surfaces changes; this can occur when a surface is added, deselected, or selected. It does not need to be run again if the display is zoomed or panned as the results are calculated on the original surface resolution, not display resolution. The overview works as follows:

1. Clear the accumulation buffers. Set a (cleared) current and previous buffer.
2. Combine the previous accumulation buffer with the current (surface) data image to generate a new accumulation buffer. Repeat this step for each surface that is active for the visualization swapping the buffers at each stage.
3. Create a color image by applying a color map to the final accumulation buffer.
4. Count the values in the final accumulation buffer to create the overview histogram.

Two accumulation buffers are necessary because a single image can not be both read and written by an OpenCL kernel.

For the focus algorithm, our accumulation buffers do not count occurrences of overlap; instead, they store a mask of which surfaces participated in the overlap. For example, if the flag was 0110, then surfaces 2 and 3 overlap in that location. The focus algorithm is:

1. Clear the accumulation buffers, except the buffer with the current results from the previous algorithm. If the number of input surfaces is odd, the active buffer will be the 2nd buffer; otherwise, the active buffer will be the first buffer.
2. For each input surface, write identity flag values into the accumulation buffer. The flag values are increasing powers of 2, where the first surface has a flag value of 1, the second a value of 2 and so forth. These values are masked using a combination of the selected level of overlap and the final accumulation surface from the first algorithm. The results is that identity flags are only accumulated at locations with the correct overlap value.
3. For non-zero values in the active buffer, determine which unique interaction group each location the corresponds to.
4. A color image is generated by combining the image generated in algorithm 1 and desaturating it at all locations except the locations that pass the previously described masking test. For locations that pass the masking test, categorical colors are displayed depending the identity flag values recorded at that location.

This algorithms where implemented with multiprocessing OpenCL kernels, which are described in the next section.

### 4.2. Kernels

Processing data with OpenCL requires the creations of kernels (GPU programs). Six different kernels were necessary for the algorithms described previously:

1. **Initialization kernel**. This kernel zeros the contents of an integer image buffer. It is used at the beginning of computation to clear results from previous computations.
2. **Surface accumulation kernel**. This kernel computes the number surfaces that have a defined value at a location in the input surfaces; it is used by the overview. The kernel works by updating the count from the previous accumulation buffer and storing in the new current buffer; these then swap places as new surfaces are processed (as OpenGL cannot read and write to the same buffer). The output is an image that contains the number of overlaps present at each location; it is further processed by both the counting kernel and the overview display kernel.
3. **Participation accumulation kernel**. This kernel computes which surfaces are part of an interaction/overlap for a given interaction level; it is only used by the detail algorithm. Like the surface accumulation kernel, this kernel uses a previous/read buffer and a current/write buffer. For input, this kernel also the results of the surface accumulation kernel and an overlap level as a mask. For example, if the overlap level is two, values will only be accumulated at positions where only two surfaces overlap. If a surface participates in the given overlap level at a location, its identity flag is combined with the existing bit-flag in the new accumulation buffer; the output is the accumulation of all

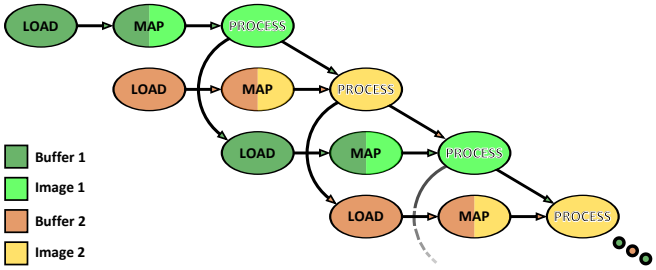



**Figure 3:** *Ordering of OpenCL operations to allow maximum throughput in visualization system; we begin loading of a data surface while the previously loaded surface is being processed. Arrows represent event dependencies: An event can not complete until all connected events have completed. Two image and two data buffers are used in alternating order for parallel storage and processing.*

| N | Processing Time ($\mu$s) |
|---|---|
| 10 | 35,602.03 |
| 100 | 338,179.28 |
| 1,000 | 3,354,251.81 |
| 10,000 | 33,782,002.28 |

**Figure 4:** *Processing Time with Synthetic Data: The overview algorithm was tested for scalability by processing surfaces repeatedly to allow timing information to be collected for very large values of N. The resulting number show that performance is linear as the number of surfaces increases. Timing data is accurate to $\pm 0.01$ $\mu$s*

these flags over the image. This output is further processed by the focus display kernel and the counting kernel.

4. **The counting kernel**. The counting kernel counts unique occurrences of values in an output buffer; its output is a 1D buffer mapping the value (index) to its count (position 0 counts how many times 0 occurs and so forth). For the surface accumulation buffer, it counts how many times a different overlap level occurs; for the participation accumulation buffer, it counts how many times a unique combination of surfaces occur. This output is used either by the histogram, the relative amount of participation, or the interaction combinations depending on view.
5. **The overview display kernel**. This kernel converts the accumulated surface counts from the surfaces accumulation kernel into a RGBA image for display using a white-blue saturation ramp. The output is rendered by OpenGL.
6. **The focus display kernel**. This kernel combines the results of the surface accumulation kernel and the participation accumulation kernel to create a RGBA image where pixels that contained a user selected number of overlapping surfaces have been recolored with categorical colors. The output of this kernel is also passed to OpenGL for display.

All calls to OpenCL are handled through an event controlled command queue. This allows simultaneous upload of data to the GPU and processing of data by using two different buffer systems, as illustrated by Figure 3.

### 4.3. Resources

The OpenCL resources required for this system consist of

- Two data transfer buffers located in pinned memory.
- Two data images in GPU memory.
- Three accumulation images in GPU memory of type unsigned int 16.
- Two display images which are acquired from OpenGL as shared resources
- One counting buffer.

The data transfer buffers are sized according to input data dimensions. The size of counting buffer is determined by the largest possible value which could be encountered; this value will be less than $2^{16} - 1$. For the data used in our first test case, which has dimensions of 3712*4316, the required GPU memory is approximately 625 MB. This value does not change as the number of surfaces used in the visualization increases.

### 4.4. Scalability & Technical Limits

The use of OpenCL places certain limits on processing:

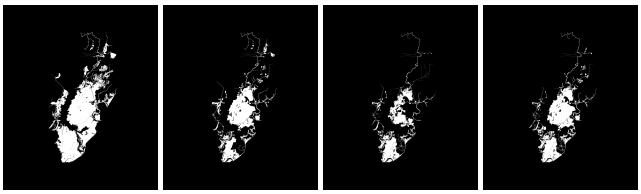
- The largest image that can processed as a single unit is determined by the OpenCL implementation's maximum image size.
- The use of 16bit integers limits the maximum number of surfaces processed in the overview phase to $2^{16} - 1$ and by the focus phase to 15. These number could be increased by using 32bit accumulation surfaces but doing so would double the accumulation buffers memory and increase the potential size of the counting buffer exponentially. A GPU-resident map structure could overcome this limitation.

One of the goals of this visualization system was scalability, to allow efficient processing of very large datasets such as from LIDAR. However, sufficiently large datasets here not available thoroughly to test the system's scalability. To address this, the first algorithm was synthetically tested by passing repeating surfaces as inputs. This allowed the first algorithm to be tested with a very high number surfaces and consequently processing a large amount of data; this is similar to how multiple LIDAR datasets would stress the system. The resulting times show that the processing speed grows linearly with number of surfaces (Figure 4).

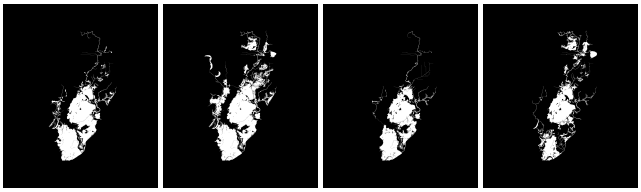
The scaling of the system, with respect to image dimensions was tested with 3 different surface sizes (~2k×2k, ~4k×4k, ~8k×8k) and found to perform composition of 10 surfaces in 10 ms, 35 ms, and 178 ms respectively. In addition, because there is no dependency between pixels of input images, large images can be segmented and the results joined to allow processing of images exceeding OpenCL's maximum size.

## 5. Case Studies

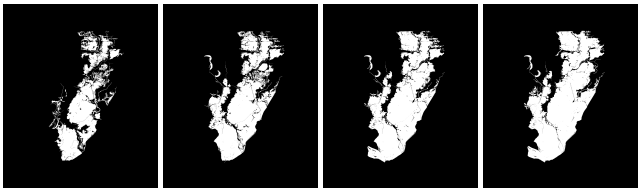
To demonstrate our approach, we present two case studies from different ground water scenarios. They demonstrate our system's efficacy on model, remotely sensed, and ensemble data.



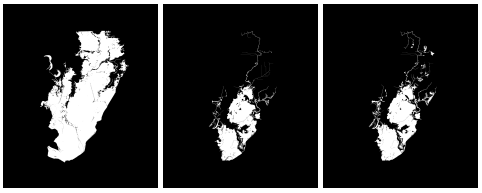
**(a)** *Surfaces from left to right: 14 day duration floods 1901 - 1931, 1932 - 1957, 1958 - 1978, Backwater Levee Study Base Condition*



**(b)** *Surfaces from left to right: May 5, 2008; May 17, 2008; May 28, 2008; 1 year frequency event*

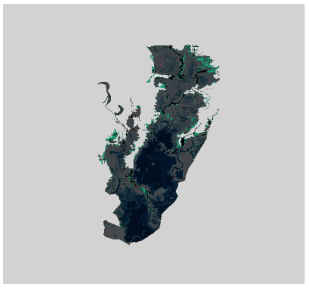


**(c)** *Surfaces from left to right: 2, 5, 10, and 25 year frequency events*



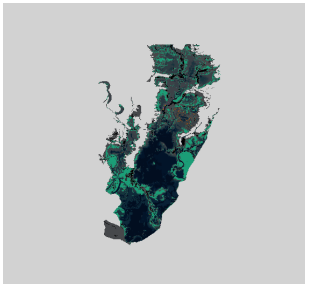
**(d)** *Surfaces from left to right: 50 year frequency event; April 16, 2008; June 8, 2008*

**Figure 5:** *The surfaces that make up the Yazoo Backwater dataset.*

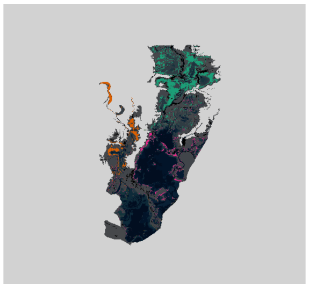


**(a)** *Areas unique to the 25-year frequency flood*

**(b)** *areas common to only the 10 and 25-year frequency floods.*



**(c)** *overlap of three layers (5, 10 and 25-year frequency floods.*



**(d)** *the overlap of the 2, 5, 10, and 25-year frequency floods.*

**Figure 6:** *Flooding in the upper regions of the project area occurs only during frequency events; this indicates short term flooding. The green surface shown in (a) is made from the 25 year frequency flood. In (b), the surface is the combination of the 25 and 10 year frequency floods. In (c) and (d), the green surface represents the combination of the 25, 10, and 5 year frequencies, and the 25, 10, 5, and 2 year frequencies respectively. Because all of the indicated green locations are covered only by frequency events it means none of them are ever flooded for a period of at least 14 days.*

### 5.1. Yazoo Backwater

The Yazoo Backwater is an approximately 1,550 square mile region between the eastern main line levee of the Mississippi River and the western Will Whittington Canal levee, bounded on the north by US Highway 82. This area has been the focus of many hydrological studies performed by the USACE (Army Corps of Engineers). The inputs for this study came from three groups: historic median annual 14-day durations floods from different time periods, a series of frequency flood simulations showing events with decreasing frequency of occurrence, and simulation data recreating the effects of the 2008 Mississippi River Flood on the Yazoo Backwater Area. The individual flood maps can be seen in Figure 5; all the data considered in this test came from hydrologic models. A quick examination of the data reveals that the 50 year flood is noticeably larger than all other events and can be excluded from further consideration. After removing the 50 year flood (Figure 1), selecting the first few levels of overlap shows that all flooding in the northern area of region occurs only in frequency events (Figure 6); this is revealed by only surface interaction made up of frequency flood events covering the entire area. This means that none of the indicated areas are ever flooded for a period of two weeks in the events considered.

The four overlap region also shows two additional areas with significant contributions. The pink ring around the central region reveals that the 5 year frequency flood has matching extents with the 1901 to 1931 14-day duration events in the central part of the basin (Figure 7). The orange area on the left hand side shows a local correspondence between the 2008 flood and the 5-year frequency flood, showing the localized severity of that event. The selection of the all but one level of overlap reveals the effects of a water control structure completed in 1958 (Figure 8). Specifically, the regions shown in pink were made to stop flooding and the regions shown in orange began to flood as a result of the water control structure.

Our system enables the detection of these regions by allowing rapid interrogation of different levels of overlap and the culling of surfaces on demand.

**(a)** *Selection of overlap regions 4* **(b)** *Areas flooded in all events* **(c)** *Areas flooded in all events but one* **(d)** *Zoom in on region of interest*

**Figure 7:** *The purple ring surrounding the dark blue center area (Figure 7a) shows the degree that the 14-day duration flood has decreased over the period from 1901 to the present. Note how it surrounds the areas shown in (b) and (c).*

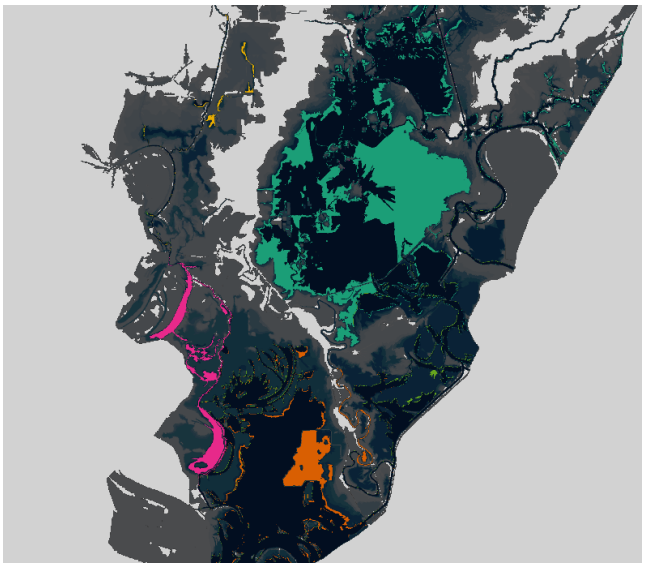


**Figure 8:** *The effects of a flood control channel completed in 1958. The green surface shows where all flood surfaces are defined except the 14 day duration flood events from 1958–1978. The green surface shows the flood surfaces previous to 1958. In 1958, a water transport channel was completed that was intended to move water from the northern part of the basin to the southern part. As a result, the indicated areas in the northern area are no longer being flooded for 14 days and thus being removed from the 14 day duration surface. At the same time, the orange areas became part of the surface due to being flooded longer.*

### 5.2. Bayou Meto

The Bayou Meto Wildlife Management Area is located off the Arkansas river southeast of Little Rock, AR, and east of Pine Bluff, AR. It is an important waterfowl management area in the southeastern united states. Seven remotely sensed flood maps from this area were used to test the visualization system with satellite imagery, instead of the model outputs used previously. The initial visualization for this data set appears to be quite noisy (Figure 9). This can partially be explained by the fact farmers in the region purposefully flood some of their fields during waterfowl season; however, the particular fields artificially flooded changes year-to-year, resulting in more or less random areas of isolated flooding in each scene. However, some insight can still be retrieved. The maximum level of overlap (Figure 9d) easily shows the primary channel of the Arkansas river and various cutoff lakes which where difficult to see in the initial image. The all but one and all but two levels of overlap (Figure 9c) reveal the extents of the two wildlife management areas in the study region, as well as the wetlands region in the north eastern part of the images.

## 6. Expert Feedback

To determine if the proposed system would be useful to experts, hydraulic engineers from the Army Corps of Engineers in Vicksburg, MS, were interviewed and allowed to interact with the visualization system and the test datasets. Feedback from the engineers indicated that their analysis of multiple interacting surface data was greatly improved—data features discovered via our system would require multiple hours of work to locate using their current workflows. In addition, due to the difference in required user time, they believe more features could be more reliably discovered with our approach. This increase in efficiency was perceived as increasing the amounts of hydrological studies performed given monetary and time constraints—less time per study increases the number of studies performed.

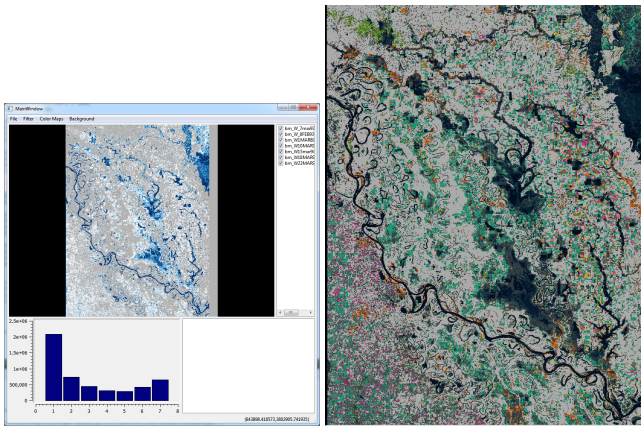
Some suggested areas where our visualization system would be of use include:

- Determination of a best fit scene from remotely sensed data to calibrate a model against.
- Comparison between the output of different models simulating the same event.
- Rapid comparison of flood extents; currently this can require hours using existing software.
- Defining domains to be modeled while minimizing potential over or under estimation.

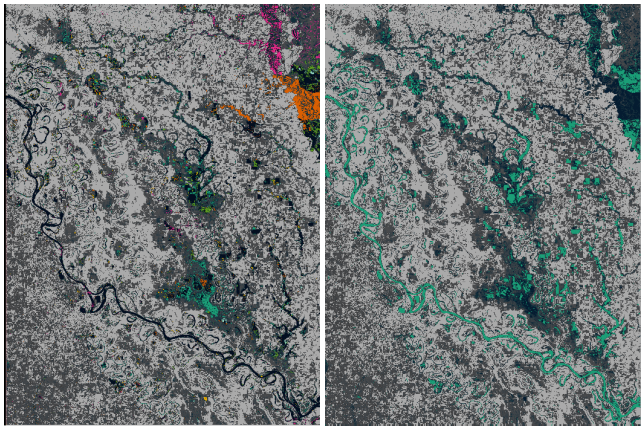
Overall, our system was found to have positive potential impact and suggestions for additional features were provided by our experts.

## 7. Future Work

One potential enhancement to the technique was requested during the interviews. The request was to extend the technique to work with pre-categorized data, where each category would be treated as a separate input. This can currently be achieved by preprocessing surfaces but one of the goals of this system is to minimize the amount of preprocessing required. In addition, extension to vector



**(a)** *Initial display for Bayou Meto* **(b)** *Areas flooded in only one event*



**(c)** *Areas flooded in all events but one.* **(d)** *Areas flooded in all events*

**Figure 9:** *The Bayou Meto dataset is dominated by surfaces with no overlap (histogram in (a)); selecting these regions results in the separate regions dominated by an event from March 1, 1989 (b, green) and noise caused farmland that is deliberately flooded. In (c), the green surfaces are two wildlife management areas in the region. The orange and pink areas in upper right hand corner are part of a wetlands area. In (d), the areas of complete agreement reveal the Arkansas river channel, lakes, and parts of the wildlife management areas that are kept inundated.*

fields by analysis of streamlines or similar markers is being considered. Finally, extending the accumulation kernels to potentially sample areas larger than a single pixel at a time when determining if surfaces agree is an avenue of potential research.

## 8. Conclusion

We have presented a new GPU-accelerated visualization system that combines overviews, region selection, and statistical data to allow rapid comparison of multiple surface agreement and conflict. This system was tested with both ensemble data and data sets consisting of multiple flood scenes. Interacting with the visualization system allows features to quickly be located in both data types. Initial consultations with field experts in flood modeling indicate that the proposed system would greatly improve analysis of such datasets, and could save significant amounts of time and money.

## Acknowledgments

The authors wish to thank the Vicksburg District of The Army Corps of Engineers for supplying the data used for these visualizations and participating in our evaluations.

## References

[AWH*12] ALABI O. S., WU X., HARTER J. M., PHADKE M., PINTO L., PETERSEN H., BASS S., KEIFER M., ZHONG S., HEALEY C., ET AL.: Comparative visualization of ensembles using ensemble surface slicing. In *IS&T/SPIE Electronic Imaging* (2012), p. 82940U. 3

[BHW06] BAIR A., HOUSE D., WARE C.: Texturing of layered surfaces for optimal viewing. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 1125–1132. doi:10.1109/TVCG.2006.183. 3

[Bok03] BOKINSKY A.: *Multivariate Data Visualization with Data-Driven Spots*. PhD thesis, University of North Carolina at Chapel Hill, 2003. 2

[CDM06] CAI S., DU Q., MOORHEAD R.: Remote sensing image visualization using double layers. In *IEEE International Conference on Geoscience and Remote Sensing Symposium 2006* (2006), pp. 249–252. doi:10.1109/IGARSS.2006.68. 2

[CDM10] CAI S., DU Q., MOORHEAD R.: Feature-driven multilayer visualization for remotely sensed hyperspectral imagery. *IEEE Transactions on Geoscience and Remote Sensing 48*, 9 (2010), 3471–3481. doi:10.1109/TGRS.2010.2047021. 2

[EVMJ*11] ELMQVIST N., VANDE MOERE A., JETTER H.-C., CERNEA D., REITERER H., JANKUN-KELLY T. J.: Fluid interaction for information visualization. *Information Visualization 10*, 4 (2011), 327–340. doi:10.1177/1473871611413180. 1

[FSL05] FORSELL C., SEIPEL S., LIND M.: Simple 3d glyphs for spatial multivariate data. In *IEEE Symposium on Information Visualization 2005* (2005), pp. 119–124. doi:10.1109/INFVIS.2005.1532137. 2

[HBW05] HOUSE D., BAIR A., WARE C.: On the optimization of visualizations of complex phenomena. In *IEEE Visualization 2005* (2005), pp. 87–94. doi:10.1109/VISUAL.2005.1532782. 3

[Hsi10] HSIAO P.-L.: *Visualization of Large Document Collections*. PhD thesis, North Carolina State University, 2010. AAI3442642. 2, 3

[JJK14] JOHNSON D. W., JANKUN-KELLY T. J.: *Efficacy of Images Versus Data Buffers: Optimizing Interactive Applications Utilizing OpenCL for Scientific Visualization*. Tech. Rep. MSU-140331-01, Mississippi State University Department of Computer Science and Engineering, 2014. 4

[PPA*12] PHADKE M. N., PINTO L., ALABI O., HARTER J., TAYLOR II R. M., WU X., PETERSEN H., BASS S. A., HEALEY C. G.: Exploring ensemble visualization. In *IS&T/SPIE Electronic Imaging* (2012), p. 82940B. 3

[SZD*10] SANYAL J., ZHANG S., DYER J., MERCER A., AMBURN P., MOORHEAD R.: Noodles: A tool for visualization of numerical weather model ensemble uncertainty. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov 2010), 1421–1430. doi:10.1109/TVCG.2010.181. 3

[Tay02] TAYLOR R.: Visualizing multiple fields on the same surface. *IEEE Computer Graphics & Applications 22*, 3 (2002), 6–10. doi:10.1109/MCG.2002.999781. 2

[WEL*00] WEIGLE C., EMIGH W., LIU G., RUSSELL M. TAYLOR II, ENNS J. T., HEALEY C. G.: Oriented sliver textures: A technique for local value estimation of multiple scalar fields. In *Proceedings of Graphics Interface* (2000). 2