Real-time Data Delivery and Remote Visualization through Multi-layer Interfaces

Vinaitheerthan Sundaram, Lan Zhao, Carol X. Song, Bedrich Benes*, Rakesh Veeramacheni, and Peter Kristof
Rosen Center for Advanced Computing, Department of Computer Graphics Technology*
Purdue University, West Lafayette, 47906, USA
{vsundar, zhao4, carolxsong, bbenes, rveerama, pkristof}@purdue.edu

Abstract

The high resolution NEXRAD Level II data provides critical information for researchers and the broader community to understand, monitor, and predict weather in a timely manner. There are several limitations in existing systems for providing easy-to-access 3D visualization of the radar data to the user community. In this paper, we present a scalable and user driven solution for near-real-time remote radar data access, processing, and 3D visualization. The system provides multi-layered interfaces for a broad range of users with different levels of services, allowing them to interactively explore data from multiple radar stations over a time period of interest. Parallel data pre-processing using the Purdue Condor pool and volume caching are implemented to help improve the system response time and scalability. The system also provides reusable radar data services and a set of access points which can be invoked by third party applications. With this hierarchical and user driven design, our system creates a rich and easy to use NEXRAD data service environment for research and education users.

1. Introduction

NEXRAD (Next Generation Weather Radar System) has been used to warn people about dangerous weather and their locations for a long time [1]. The full national feed of NEXRAD (also known as Weather Surveillance Radar) Level II data is available in real time from the National Weather Service (NWS). The data provides radar reflectivity, Doppler radial velocity, and spectrum width at high temporal and spatial resolution, which are critical for understanding, monitoring, and predicting severe weather and flooding events.

Several radar data visualization tools and systems exist. However, despite the availability of 3D information in the new generation radar data, the data is most commonly displayed as 2D images (for a single radar or overlapped using transparency for the data from multiple sites), 3D point clouds, or isosurfaces. The resulting images provide limited information about the movement of the frontal systems, as viewed on web-sites such as www.weather.com, www.wunderground.com or when viewed on TV. These are usually enough for a simplified view but cannot accurately represent details. Moreover, most of the existing systems have limitations such as (1) not being able to handle large amounts of data from many stations over a long period of time, (2) not allowing users to directly interact with the data, (3) not being available to the general public because the tools have complicated interfaces and/or require advanced hardware and domain expertise, or (4) not providing any access point for third party applications. We developed a system to address these limitations, and more importantly, cater to a broad range of users with different levels of computing expertise and for various use scenarios.

There are several challenges in developing such a system. First, the radar data streams at ~50 MB/second and is of large volumes. Most traditional data management systems are not effective in handling this type of data efficiently. Second, the raw radar data is stored in a customized compressed format upon receiving and cannot be read directly by existing radar data libraries. Third, it is computationally intensive to process the data, especially when users need to analyze data at large geospatial and temporal scales. Moreover, there is a special hardware (accelerated GPU) requirement if high-quality 3D visualization needs to be produced. It is a challenging task to design a system that not only offers powerful interactivity with large volumes of real time streaming data but also scales well to serve many users simultaneously.

Aimed at addressing these problems, we designed and developed a versatile environment for NEXRAD level II data management, processing, and visualization using a user driven approach. This system adopts a
Our overarching goal is to create scalable NEXRAD data analysis and visualization services, and create an extensible architecture that supports distributed computation and visualization applications developed by partner institutions. Towards this goal, we have designed and developed a service oriented prototype system as shown in Figure 1.

The core of the system consists of three components for data access, processing, and 3D visualization. All components expose service access points that can be invoked by third party applications. This design allows us to distribute data processing and graphics rendering workload between remote and local systems in a flexible manner. For example, these core components make it easy for us to provide three types of user interfaces targeting the user groups discussed earlier. For learners and casual users, we provide several preconfigured web gadgets that present near real-time 3D radar data visualization with high system scalability. These gadgets hide the functional complexities and present simple interfaces to the users. They are easy to use and meet the needs of users at the starting point of a learning curve. For the expert users, we provide a downloadable application that can be easily installed on a GPU-enabled computer. The multi-layer interface design which has been proven to provide a feasible and flexible interface solution to meet the different needs of various end-users [2, 3]. In a multi-layer interface design, the users are divided into several groups based on characteristics such as domain expertise, feature requirements, and/or administrative roles. The system provides interfaces at several layers with different levels of complexity. Each layer provides different functionalities that are appropriate for the corresponding groups of users. By adding more layers, the flexibility of the software system in supporting various users increases. It also enhances learning by providing appropriate levels of complexity so that users can master the functions and become productive in using the tools with a small learning curve.

Through our interaction with end users, we observed three groups of NEXRAD II data users based on their access patterns: (1) Expert users: a small group of users who need to perform in depth investigation using the radar data. They need full access to a large amount of radar data and visualize it in 3D. This class of users includes mostly academic researchers, emergency management agents, and other professionals in the related fields. (2) Learners and casual users: a large group of users who want to access the data and visualize it for educational, personal and other interests. They need quick and easy access to the data and visualization of past severe weather events as well as the most recent radar captures. They have limited knowledge about the scientific domain. The requirements for data resolution and 3D interaction are much lower compared to the expert users. This class of users includes K-12 and college students, as well as the general public. (3) Advanced users: a small group of users, for example, graduate students, researchers from other scientific domains, who want to explore and evaluate the data but do not have special purpose accelerated graphics hardware or software installed. They often wish to use the data for a short term and their requirements fall between the other two user groups. Note that these user classes are not mutually exclusive. For example, an expert user may act like an advanced user when accessing data from a remote location such as in a scientific conference, or like a casual user when teaching in a classroom or a workshop. In all of these use cases, however, the common user need is the capability of near-real-time data access and visualization.

Our strategy was derived from the different characteristics of the user groups described above, mainly their data access pattern, data access frequency and the level of interactivity. The system provides multiple user interfaces to support different usage scenarios. These interfaces offer different levels of data and visualization functionalities at varying scales. To support these user interfaces, we implemented a set of services for radar data management, access, processing, and visualization in the backend system in such a way that multiple access points can be provided for accessing these services remotely. To achieve fast responsiveness and near-real-time data access, we used techniques such as parallel data preprocessing on the TeraGrid resources and volume caching [4]. This system integrates near-real-time hardware accelerated 3D visualization and distributed data management and processing technologies, allowing remote access and analysis of the data in a rich learning environment. It is versatile and can be incorporated in different research and learning environments. The core services, which we will describe in the “Services and Access Points” section, will also enable integration with public information dissemination mechanisms.

In the following sections, we first present the overall system design, and then describe the multiple user interfaces and their implementation. We will describe in greater detail the design and implementation of the data access, processing, and visualization components and services in section 4, followed by a discussion of related work and our future plan at the end of this paper.

2. Overview of System Design

Our overarching goal is to create scalable NEXRAD data analysis and visualization services, and create an extensible architecture that supports distributed computation and visualization applications developed by partner institutions. Towards this goal, we have designed and developed a service oriented prototype system as shown in Figure 1.

The core of the system consists of three components for data access, processing, and 3D visualization. All components expose service access points that can be invoked by third party applications. This design allows us to distribute data processing and graphics rendering workload between remote and local systems in a flexible manner. For example, these core components make it easy for us to provide three types of user interfaces targeting the user groups discussed earlier. For learners and casual users, we provide several preconfigured web gadgets that present near real-time 3D radar data visualization with high system scalability. These gadgets hide the functional complexities and present simple interfaces to the users. They are easy to use and meet the needs of users at the starting point of a learning curve. For the expert users, we provide a downloadable application that can be easily installed on a GPU-enabled computer. The
application connects to the data remotely through data service access points, and renders the 3D graphics locally on user’s computer. It allows the user to interact with 3D visualization of the data covering multiple radar stations over a period of time specified by the user. This user interface focuses on the needs of the expert users by providing a maximum level of functionalities. It scales fairly well because graphics rendering is performed on local graphics hardware. For other users who do not have local GPU-enabled hardware or who simply cannot or do not want to install any software, we provide a web browser interface through which users can access, visualize and interact with the data while all the processing and rendering happen on the remote servers. In this case, the scalability depends on the remote hardware resources. For detailed performance assessment, we refer the readers to our earlier work [8]. With this hierarchical approach, our system is designed to cater to varying user needs at different scales required. Our system is currently in research prototyping stage.

3. Multi-Layer User Interfaces

In this section, we explain in detail the design and implementation of the different types of interfaces supported by the distributed radar data environment.

3.1. Web Gadgets

The Web 2.0 technologies such as AJAX, Google gadgets, social networking applications, provide an effective and powerful platform for rapid dissemination of scientific tools and services to a wide audience, including the K-12 and college students [5, 6]. For example, there are 31 Google gadgets that have more than 500,000 users, and 146 Facebook applications that have more than 500,000 active monthly users.

Leveraging these technologies, we have developed “LiveRadar3D”, a Google gadget that uses an embedded flash player to stream an animated movie of the 3D views of the past few hours of radar data. General users can add LiveRadar3D gadget (Figure 2) to a web page or their computers if they have Google Desktop installed. Based on the zip code selected by the user, the gadget maps it to one of the seven regions that have been predefined over the entire United States. The animated movie shows the 3D visualization of the current radar data from all the stations in that region, which includes 3D camera views, scaling, and 3D rotation. Using preprocessing and volume caching to generate the animation movies as soon as new data arrives, this solution is scalable, has a very fast response time, and delivers the latest radar data in near-real-time. In addition, we plan to implement several more gadgets with preconfigured functionalities such as 3D visualization of past severe weather events, as well as visualization containing different color maps.

Apart from Google gadgets, the same web application system could easily be adapted to other web 2.0 technologies. For example, similar web gadgets can be developed using the Facebook application framework and added to the Facebook application directory [6]. Our system architecture supports a wide range of mobile technologies that could put scientific data and visual representation into the hands of millions of people with modest level of infrastructure support.

Although the data resolution and level of user control through web gadgets are limited, this lightweight web application serves the needs of the majority of entry-level users ranging from personal interests on real-time weather events to learning. These web gadgets can easily be integrated into course materials, learning and teaching modules.

Since the animated movies are generated offline using preprocessing and data volume caching (described in Section 4), the scalability of this solution is mainly determined by the performance of the web server to stream the videos. The videos are generally a few megabytes in size, in which case thousands of concurrent users can be supported by modern web servers such as Apache2 [7].
3.2. Personal Desktop Client

For researchers and emergency management personnel and other expert users who need full interactive capabilities and high resolution visualization, we developed a LiveRadar3D desktop client using our data processing and access services. It was written in C++ and built on a number of libraries including RSL, Cg, OpenGL, GLUT, and GLUI. An example visualization of all stations in the entire United States is shown in Figure 3. Since the libraries used are platform independent, the code can be compiled and installed on both Linux and Windows running on a computer equipped with a standard GPU, which is inexpensive and easily available nowadays. The graphical user interface of the client allows a user to select a geographic region from a map and the time period of interest. The tool then connects to the data access interface to query and fetch the corresponding volume data remotely. Upon receiving the processed data, it will render the data using the local GPU and display the visualization over the map.

The advantage of using the desktop client is that it allows the user to access a large amount of data that may cover the entire United States and span a long time period. The user can also interactively manipulate the 3D visualization by zooming and rotating it and by changing the transfer function to study the details of a storm structure or identify potential areas of turbulence from different directions. By distributing the rendering load to the client machine and archiving preprocessed data volumes on the server, the amount of on-demand computation is reduced significantly. As a result, this solution scales well to serve the targeted community which is much smaller compared to the group of learner/casual users.

3.3. VNC Enabled Web Interface

For those users who are interested in exploring the data without downloading the client program or who do not have a GPU-enabled computer, they can access, analyze and interact with the data from a web browser. The 3D data view is generated by our remote visualization tool and presented to the user in a Virtual Network Computing (VNC) applet viewer embedded in the browser [30]. The visualization tool runs on the remote server and functions similar to the LiveRadar3D desktop client except that it supports multiple users. This is a convenient way to provide remote advanced interactive as well as collaborative visualization of NEXRAD II data using our end-to-end system without requiring special purpose graphics capabilities on user’s local computer. However, this solution has its scaling limitation in our current prototype because all the rendering and visualization are done on a central server. We are currently investigating ways to scale up this solution, including building a server farm.

3.4. Third Party Application Development

In addition to the three interfaces for end users of the radar data visualization described above, we also provide fine-grained service access points that can be used by third parties who wish to develop radar data related applications. For this purpose, the backend system implements multiple layers of service interfaces that can be invoked by other programs, thus creating opportunities for other uses of the radar data and services that we provide. A detailed discussion on
accessing data from backend system is in Section 4.2. For example, a weather prediction application can use the data access interface to download uncompressed NEXRAD II data and feed the data to a mathematical model. A user can also develop applications that provide custom 3D visualization of the most recent radar data by fetching the pre-processed data volumes from the data repository. It is also possible to develop more advanced applications that invoke the services for data dissemination, aggregation, and visualization and combine the data with high-resolution models such as Weather Research and Forecasting (WRF) with tracer transport, dispersal, and deposition components via real-time data assimilation.

4. Services and Access Points

In our previous work, we have developed an end-to-end on-demand data processing system [8]. The system fetches data from the repository, processes, and renders it each time a user submits a request. However, this on-demand system is not scalable to multiple users and becomes slow when a large amount of data must be processed. We have since extended our previous work and implemented a prototype of a scalable service oriented system to support users of all levels as well as third party applications. The new system provides more services and access points, pre-processes data as it is received, and caches the 3D data volumes in our data management system for easy access and reuse. A detailed view of the data flow within this system is illustrated in Figure 4. The system consists of three main components: data management, processing, and visualization. In this section we will describe the design and implementation of each component.

4.1. Overview of NEXRAD II Data

There are 158 operational NEXRAD radar systems deployed throughout the United States and at selected overseas locations. The maximum range of the NEXRAD radar is 250 nautical miles [1]. The NEXRAD Level II data are collected as the radars go through a programmed set of movements, which involve a continuous rotation over 360° in azimuth and a simultaneous increase in elevation by 1° to 3° per complete sweep [9]. The spatial resolution is one kilometer for reflectivity and 0.25 kilometer for velocity and spectrum width in range, and one for all three fields in azimuth. The radar makes up to 14 scans in elevation ranged from 0.5° to 19.5° determined by the Volume Coverage Patterns.

The data stream is distributed from NWS and produces the raw data on a continuous basis. The radar data files vary in size from a few megabytes to tens of megabytes each, depending on the weather condition. The temporal resolution is 4-5 minutes in severe weather vs. 9-10 minutes in calm weather. The raw data is partially compressed with bzip2 algorithm before it is stored at the distributor sites. The sizes of the compressed files vary from a few hundreds of kilobytes to a few megabytes depending on weather condition.
4.2. Data Management and Access

We developed a set of services for easy access to the radar data from a Storage Resource Broker (SRB) based data management system [10]. There are three main challenges in managing the streaming radar data and providing easy access to users: (1) The radar data is received continuously at near real-time via the Local Data Management (LDM) software and, as a result, the data management system needs to be able to efficiently detect and manage the latest data immediately after it is received. (2) The raw files are in a unique binary compressed format which cannot be read directly by existing off-the-shelf radar data libraries such as NASA TRMM Radar Software Library (RSL) [11]. (3) There is a lack of easy ways to access the data in a timely manner to support time-sensitive applications. Users often download data using FTP or HTTP.

In order to manage this real-time streaming data efficiently, we defined each radar station as a special SRB shadow directory object. As shown in Figure 4, new data are automatically registered under a shadow directory managed by a SRB data grid on the TeraGrid. We keep four types of data in the repository: (1) the raw compressed radar data from the recent month are stored online; (2) the data from the entire year are archived in a tape storage system; (3) the partial data volumes for the past year generated by the data preprocessing component (as explained in Section 4C); and (4) the data that cover severe weather events are also available. All these data can be easily accessed through the data management system.

There are several ways to query and retrieve the data. First, a THREDDS (Thematic Real-time Environmental Distributed Data Services) application server and an OPeNDAP (Open-source Project for a Network Data Access Protocol) application server are integrated with the SRB middleware to provide a dynamically generated radar data catalog and transparent remote access via THREDDS and OPeNDAP protocols [12, 13]. Users can also access the data remotely using various SRB clients or several existing radar data visualization tools including IDV [14]. To make the radar data available to a wider user community, we further developed a set of radar data access interfaces that retrieve the desired radar files and convert them into usable format based on the radar station name and the time period provided by the user. It can also be used to retrieve variable values such as reflectivity and radial velocity from the compressed data files. The interfaces are a set of open source Application Programming Interfaces (API) provided as a C++ library. It was developed using the SRB C library and the RSL library. Using these APIs, a user can directly access any radar data file available in our repository.

Because of the wide area coverage, high temporal and spatial resolution of NEXRAD observations, and their availability in near real-time via high speed TeraGrid network, this data set is unique in its ability of providing initial conditions and evolution constraints for numerical weather prediction models, such as providing important constraints when assimilated into models and analyses [15]. Our easy-to-access data services allow researchers to utilize the NEXRAD data in models and tools without having to learn the details of how to find and convert the data.

4.3. Data Processing

In this section, we first describe the steps for data processing. We then describe partial volumes that can be created independently and merged later into a render-able full volume. The partial volumes have some key attributes that allow us to use pre-processing and caching to increase the scalability of the entire system.

4.3.1. Generation of 3D volumes. To ensure an efficient volume rendering process in the rendering stage, all data is resampled in a rectilinear grid structure which contains a collection of cells arranged on a regular lattice [16]. A bounding box that contains data from all the sites was generated first. There are more data points distributed in the xy-plane than in the z-direction (the elevation direction) in the grid. Therefore, the grid (Figure 5) has a non-uniform structure such as 256 by 256 by 128, which provides the grid with 128 layers of xy-slices, each consisting of a 2D uniform grid of 256 by 256.

![Figure 5. 3D 256x256x128 grid structure (left) and bounding box.](image)

For each site, the RSL library stores the data in the local spherical coordinates (azimuth, elevation and range) with the origin at the location of each radar station. To integrate the data from the multiple sites, a
global coordinate system is used to align the data. Since radars at different locations are operated at different times and rates, the data from one site are collected at a time and rate different from another site. We perform interpolations between two time stamps to reduce the temporal aliasing.

When the reflectivity values are resampled at each grid cell into the predefined 3D array, the volume is constructed. It is easy to conceive that, in partially overlapping regions, values obtained from more than one site are likely to be sampled to the same cell, which means some cells have duplicate values, and some cells only have a single value. To eliminate the potential error resulting from the redundant data sampling, we calculate the sample average. Since the distribution of the sample values may still be sparsely populated in the volume space, vertical interpolation is applied to fill the gaps. More details are available in [17].

4.3.2. Parallel Data Pre-Processing and Partial Volume Caching. In our previous on-demand data processing system, every time the user submits a request, the backend system will fetch the related data from the SRB repository, process the data, and then render it. Due to the large data volumes and intensive computation involved in preparing the radar data for rendering, it is not efficient nor scalable if the data selected by each user needs to be processed on demand after being retrieved from the data repository. It is a primary goal to reduce the delay caused by data processing in this paper.

We observed that for a given time interval the partial volumes for individual radar stations have the following properties: (1) the partial volumes are independent of each other and can be constructed in parallel. All individual station volumes can then be merged to form a full volume for rendering. (2) For a single station, the partial volumes corresponding to a time interval are independent of the other intervals and therefore can also be computed in parallel. (3) The size of partial volumes is two orders of magnitude smaller than the raw data from which they are generated. (4) Partial volume construction is computation intensive if done sequentially. (5) Unlike the full volumes which differ based on the set of stations included, partial volumes computed for individual stations per interval are reusable. These computations are well suited for Condor technology [18].

As a result, we developed a preprocessing module that monitors the arrival of new data and then processes them into partial volumes immediately using the TeraGrid Condor resources (Figure 4). The data for each station are accessed and processed in parallel as multiple Condor jobs to an 812 8-core Dell 1950 system with various combinations of 16-32 GB RAM.

As shown in Figure 4, the partial volumes are cached for each station which can be retrieved by client applications remotely such as our personal desktop client and used for rendering. In addition, the data volumes generated are archived in the SRB for the past year which can easily be retrieved using our data accessing interfaces. By preprocessing the data as they arrive and caching the data volumes, the visualization application can directly fetch the partial volumes for the target region and merge them into the full volume for rendering. The merging and rendering time is insignificant compared to the computation of partial volumes. As a result, the response time to each user request is improved significantly, providing almost instantaneous visualization.

Three types of data processing interfaces are available for remote applications: (1) the partial volume generation interface retrieves the data from the data management system, decompresses it and processes it into partial 3D volume. (2) The merging interface combines the 3D volumes from all sites and creates the full 3D volume for each time interval. The granularity of parallelization for the processing job can vary from a single station to tens of stations and configured based on the hardware capability and the size of input data. The merging interface takes as input a set of stations and merges the partial volumes of those sites into a renderable full volume. (3) The decompress interface retrieves the raw data from the repository for a specific site and decompresses it.

4.4. Data Rendering and Visualization

Texture-based volume rendering is used to display the high-quality visualization of the weather data. In object-order volume rendering, a stack of 2D parallel polygonal slices are used to sample the 3D scalar field of radar samples. Once the 3D texture is stored in GPU, the 3D slices cut the texture and the texture mapping assigns color to each individual pixel. The slices are rendered from back-to-front order to assure correct alpha blending.

One problem we faced is the color mapping for the reflectivity variable in order to interpret it in some reasonable manner. We used the standard color mapping since most users are familiar with it. In this mapping, each value of the radar data is assigned a rainbow-like color from a predefined one-dimensional look-up table. The look-up color table is similar to the one used in the NWS web page and is widely used for radar reflectivity data [19].
Figure 6 shows four images rendered at different timestamps using a dataset from scanning a 24-hour supercell storm on March 12, 2006, in the Midwest region of the United States. The images are rendered when the view direction is tilted approximately 20 degrees relative to z-axis. It shows the developed transfer function window for red, green, blue (RGB) and alpha channels representing the NWS look-up color table [19]. All rendering was done using the NVIDIA Cg shading language. The simulation was carried out on a Windows desktop equipped with a 3.20 GHz Pentium 4 processor, 2.0 GB of main memory, and an NVIDIA Quadro FX 3500 graphics card with 256 MB of video memory. The resulting image resolution is set to 796x532 for all the experiments.

The data rendering and visualization services provided by the system allows easy-to-use 3D camera manipulation such as free rotation, scaling, animation, interactive color mapping function and color set mapping. The VNC client or desktop client users can access these services through the GUI provided by the desktop tools. To support the interactive transfer function, we created a transfer function window, allowing users to create and modify transfer functions to their interest. The created or modified transfer functions are then applied to map the data onto the appropriate color and opacity.

The Google gadget client accesses a pre-configured visualization service which generates rendered images for specified radar stations at a specific timestamp with customized requirements on 3D camera view, 3D rotation, and scaling. It can also concatenate a series of rendered images and provide an animated movie as the output format. The visualization services are mapped to the different parameters of the back end visualization capture program which accepts as input variables such as flags for view port type, color map, image size, and scaling factor. With such interfaces, it is easy for third party programs to build customized applications enabled by the remote services provided by our system.

5. Related Work

We discuss related work with respect to each of the three interfaces described above. To the best of our knowledge, our system is unique as a versatile solution that provides multi-level interfaces for accessing and visualizing NEXRAD II data and caters to different user expertise levels.

Since visualization of weather radar data is crucial for the understanding and accurate forecasting of weather related events, it has been an important research area for many years and several useful tools have been developed. Integrated Data Viewer developed by Unidata is a well-known multi-purpose tool [14]. It can read different data formats including Doppler Level II and Level III radar data, satellite imagery, gridded data, and NOAA National Profiler Network data. The display is in 2D and 3D fashions. The NOAA NCDC Java NEXRAD Viewer and Data Exporter can load level II and Level III radar data into
an OpenGIS-compliant environment [20, 21]. The LEAD (Linked Environments for Atmospheric Discovery) portal provides meteorological data, forecast models, and analysis and visualization tools as the weather evolves. It uses Integrated Data Viewer for visualization. [29] The Interactive Radar Analysis System (IRAS), a part of the Collaborative Radar Acquisition Field Test (CRAFT) project, can read and display Level II radar data via the Internet in real-time [22, 23]. There are several limitations in the existing tools. Unlike our desktop client which can handle data from many time steps and stations, most tools only handle data generated at one time step or from one station at a time. Most of these tools are written in Java for platform-independence and as a result, may suffer in performance compared to our desktop client written in C++. Moreover, most tools display 2D or simple 3D point clouds. Our personal desktop client provides a full 3D interactive representation instead.

Remote visualization presents a significant challenge. One successful application is the nanoHub that uses VNC for remote visualization of nanostructures [24]. The application uses very small data that are fit into the GPU that renders them. The GPU is immediately released so different user can use it. The use of small datasets allows for rapid change of usage of resources that is difficult for large interactions with huge datasets. Another application is the web-based interactive volumetric rendering, a part of the TeraGrid visualization gateway, which enables different applications to use the TeraGrid visualization resources [25, 26]. This application uses CPU-intensive parallel ray-casting technique as opposed to texture slicing used in our system. For 3D interactions, parallel ray-casting can be slower as texture slicing allows volumes to be directly rendered by GPU. The application, however, can potentially be a third party client for our data accessing and processing service access points. The application can obtain the full volumes and render them remotely.

Web gadgets for Doppler radar have been developed and are quite popular among general public [27, 28]. These gadgets provide temperature and warnings about important weather events like storms in addition to the Doppler radar images. Currently, these gadgets do not have 3D visualization capability. The animation is displayed outside the gadget on a separate webpage and at the granularity level of either the whole nation or a single station. In comparison, our gadget “LiveRadar3D” has 3D visualization capabilities and plays animation inside the gadget at the granularity of tens of radar stations covering the region of the user’s interest.

6. Conclusions

In this paper, we described our design and implementation of a versatile environment for NEXRAD Level II data management, processing, and 3D visualization. It uses a multi-layer interface design to provide scalable radar data visualization functionalities to a broad range of users with different levels of services. Our contribution includes:

- A set of reusable data services and multiple access points, supported by the TeraGrid resources, which provides radar data retrieval, preprocessing, and rendering in near real time.
- Parallel data pre-processing using the TeraGrid Condor resources and partial volume caching which improve the response time and system scalability.
- A personal desktop client that provides fully interactive 3D visualization of NEXRAD Level II data at large spatial and temporal scales.
- An integrated end-to-end backend system that connects radar data retrieval, processing, remote rendering, and 3D interactive visualization.
- A scalable system that supports users at different skill and interest levels.

For future work, we plan to explore the spatial and temporal characteristics of the data that is arriving in nearly real-time. Techniques will be developed to compress empty areas (as there is no cloud) and store full areas (as there is a supercell, hurricane, or simple cloud) in highest levels of details. Also, the data has very high temporal coherence as the clouds are moving relatively slowly compared to the speed of radar scanning. We would like to create a hierarchical data structure that reflects both the spatial and the temporal coherence. Some viable options are Oct-tree or BSP. We are also exploring providing an interface that supports queries based on certain thresholds within the radar datasets as it could potentially be useful for several users.

7. Acknowledgment

The authors wish to thank Gary Bertoline and Matthew Huber for their discussions and suggestions that helped the design and development of the system. We also appreciate help of Yi Ru who has developed an early version of the visualization system. The Level II Doppler radar is provided in real-time to Purdue’s Atmospheric Measurement and Prediction Consortium in partnership with National Weather Service utilizing software developed and maintained by Unidata.

8. References