Virtual Watershed Visualization for the WC-WAVE Project

Chase Carthen, Thomas J. Rushton, Nolan P. Burfield,
Christine M. Johnson, Aaron Hesson, Daniel Nielson and Bryan Worrell
University of Nevada at Reno, Reno, NV 89557 USA

Donna Delparte, Tucker Chapman and W. Joel Johansen
Idaho State University, Pocatello, ID 83209 USA

Roger Lew, Nicholas R. Wood, Mathew Ziegler and John W. Anderson
University of Idaho, Moscow, ID 83844 USA

Sergiu M. Dascalu and Frederick C. Harris Jr.
University of Nevada at Reno, Reno, NV 89557 USA
Fred.Harris@cse.unr.edu

Abstract

The platform discussed in this paper, Virtual Watershed Client, provides a tool that allows researchers, students, and stakeholders to observe and analyse both geospatial datasets and theoretical model data. The system created ingests the geospatial data and can create three-dimensional terrain, satellite imagery, shapes, and model data. This was done using the Unity3D game development engine and libraries to work with geospatial data such as GDAL and NetCDF. These tools that are developed are used by researchers, most specifically hydrologist for the purpose of this paper, to interact in a three-dimensional environment with their gathered data. The design of the Virtual Watershed Client, as well as its currently implemented analysis and visualization tools are discussed in detail throughout this paper.

Key Words: Visualization, geospatial, model data, hydrology, virtual watershed platform, virtual watershed client, terrain.

1 Introduction

Today there is a need to visualize and analyze datasets captured by sensors in an environmental watershed and produced by biophysical models. This need stems from the watershed scientists; the ability to view their collected and model data on an actual terrain is beneficial. Having the ability to see the peaks and valleys of data values line up in accordance to actual peaks and valleys of the terrain is just one example of the usefulness visualizing the datasets. Researchers use sensors to provide information about the surrounding environment of the watershed and they allow the researchers to run models to analyze the data captured from these sensors. Researchers use this acquired data to develop models that effectively reflect the environment of the watershed. These models can then be used to predict water runoff, precipitation, snow melt, underground water flow, and other environmental factors within a watershed. Models consist of many input and output variables that require visualization and tuning. There are several existing tools used to visualize this data and provide a researcher sufficient information to modify variables in their models. These tools include QGIS [30], ArcMap [6], and other Geographical Information System (GIS) and non-GIS programs. However, most of these tools lack the ability to visualize and interact with data from a first person perspective, that is to say viewing the watershed as if you were actually there in person. The Virtual Watershed Client [4] is currently being implemented and has been designed to provide researchers a 3D analytical tool to visualize model data. Researchers will be able to provide data to a web service, the Virtual Watershed Platform, and the Virtual Watershed Client can then be used to visualise this data. The Virtual Watershed Platform uses the Open Geospatial Consortium (OGC) services for serving data [28]. The client uses libraries such as the Geospatial Data Abstraction Library (GDAL) to parse any data that gets delivered to the Virtual Watershed Client [10]. Researchers are then able to observe this visualization to evaluate the outcomes of their model runs. With the modern gaming development engines, and the processing power of computers fusing a game atmosphere with large data manipulation is possible. The current implementation of the Virtual Watershed Client was built using the Unity game engine [36]. The Unity Game Engine was chosen because it allows our platform to be published on many potential platforms, such as Windows, Mac, Linux, and the web browser.

The capabilities of researchers to gather data in their respective watershed environments has coincided with the ability to aggregate and analyze that data in meaningful ways. The authors of [29] discuss methods of determining the optimal locations for dam placement in order to harvest water. They

do this by analyzing a multitude of data maps overlaid onto a Digital Elevation Model (DEM). In [23] the authors describe a toolkit built for hydrological modeling using a geographic information system such as ArcView GIS. Their system, the Automated Geospatial Watershed Assessment tool (AGWA) is capable of executing and then visualizing results inside of ArcView GIS from models such as the Soil and Water Assessment Tool (SWAT) and Kinematic Runoff and Erosion Model (KINEROS2). The virtual watershed implemented in the Sevier River Basin [3] was designed for the purpose of improving operation of river and irrigation canals. To do this, a real-time representation of the watershed was created using remote sensors that constantly stream data to a server, allowing for rapid decision making. Finally, in Advanced Techniques for Watershed Visualization [1], the authors discuss using hydroshading algorithms on NASA's Shuttle Radar Topography Mission DEM datasets in order to generate high quality 3D models of the watersheds that the the datasets pertain to.

The following discussion expands upon the details given in the paper Design of a Virtual Watershed Client for the project [4]. Section 2 discusses the overview of the funded project, and how the Virtual Watershed Client fits into that project. Section 3 explains the models used by the watershed scientists on the funded project. Section 4 covers the details of the design and implementation of the Virtual Watershed Client. Next, we demonstrate two separate watersheds in Section 5, those two watersheds being located in Dry Creek and Lehman Creek. Finally, we present conclusions and future work of this project in Section 6.

2 Project Overview

The Western Consortium for Watershed Analysis, Visualization, and Exploration (WC-WAVE) [38] is a tri-state consortium composed of researchers and students in Idaho, Nevada, and New Mexico. Its overall aim is to study the localized impact of climate change on high-mountain catchments. The WC-WAVE project is comprised of three components: Watershed Science, Visualization and Data Cyberinfrastructure (CI), and Workforce Development. Participants in these three components are collaborating to better understand interactions between precipitation, snow-pack, groundwater flow, and other watershed properties within mountain catchments. To enhance the collaborative nature of this project and future work, as well as to promote data exploration and analysis, one of the main outcomes of the WC-WAVE project is a Virtual Watershed Platform (VWP). The VWP framework will provide data access and visualization through individual workstations (desktop), web-based environments, and advanced interactive 3D environments, including stereoscopic, and immersive CAVE and Virtual Reality environments. The VWP will also allow a user to run specific hydrologic models and visualize the results. During the preliminary meetings for this proposal, project team members put together the design for the Virtual Watershed Platform. Figure 1 illustrates this VWP design.

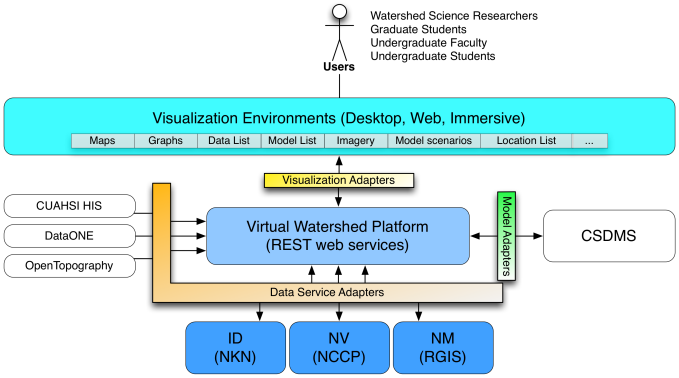


Figure 1: Diagram illustrating the relationships of proposed project components; virtual watershed platform, data management services, and portals. The Virtual Watershed Client is on the Visualization Environments block, which gets data required from the Virtual Watershed Platform. The data to the VWP comes from multiple locations. Then all data transfer is done through adapters

To implement and test the VWP, project participants selected four watersheds throughout the tri-state region to focus on: Dry Creek, Reynolds Creek Experimental Watershed in Idaho, Jemez watershed in New Mexico, and Lehman Creek in Nevada. These watersheds were chosen partly for their unique characteristics. For example, Dry Creek and Reynolds Creek provide an ideal environment for studying and developing tools related to snow pack, while the Jemez watershed is more suitable for vegetation studies. These watersheds were also chosen for their extensive data networks. Since the models within the VWP call for several different forms of data, watersheds with adequate datasets and instrumentation were required.

Members of the Watershed Science component of the WC-WAVE project selected four main models to include in the VWP: ISNOBAL [18], PRMS [35], ParFlow [20], and DFLOW [34]. The ISNOBAL (image SNOw mass and energy BALance) model is used to predict snowmelt and runoff. PRMS (Precipitation Runoff Modeling System) simulates the hydrologic process. ParFlow (PARallel FLOW) model is useful for modeling soil moisture and the interaction between surface and sub-surface flow. The DFLOW model is used for simulating river channel flow and hydrologics. These datasets are later explained in Section 3.

3 Data Models

As stated, the WC-WAVE project uses four primary models for conducting watershed science. Out of the four models ISNOBAL is the most established in VWC. During the development of the VWC, ISNOBAL datasets were most readily available for testing and development purposes,

therefore features were designed around the data provided from this model. Section 3.1 describes this model in depth, and discusses one of the watersheds in the project that contains ISNOBAL data. The other models used in VWC are represented in the same method as ISNOBAL, and are described in Section 3.2.

3.1 Snowpack Modelling

Several models have been developed for the purpose of predicting snowpack properties over the course of a season. SNOW17 [2] was designed as part of the National Weather Service's River Forecasting System, and is still in use today [8]. The SNTHERM [13] and SHAW [7] models accurately simulate snow properties at a point, while the Utah Energy Balance (UEB) model [33] and the United States Geological Survey's Precipitation-Runoff Modeling System [15] can be applied as distributed simulations over small areas. SnowModel [16] utilizes input meteorological conditions along with wind conditions, vegetation and surface energy exchange to simulate snow depth and water equivalent. The ISNOBAL model [19] is part of our current implementation, as well as the Image Processing Workbench (IPW) suite of software tools [37, 17].

ISNOBAL is effective at predicting snowmelt and runoff when applied to a range of watershed sizes (1 to 2500 km2) as well as temporal ranges (one week to an entire season). When run through IPW, ISNOBAL requires three different types of input: an initial conditions image, precipitation images, and input forcing data images. The same initial condition image can be used for an entire model run and is comprised of seven bands: elevation, roughness length, total snowcover depth, average snowcover density, active snow layer temperature, average snowcover temperature, and percentage of liquid H_2O saturation. Precipitation images consist of four bands: total precipitation mass, percentage of precipitation mass that was snow, density of snow portion of precipitation, and average precipitation temperature. These bands are only included for times of precipitation events (i.e. storms). Individual input forcing data images are required for each time step (typically three hours) throughout the model run, and are made up of six bands: incoming thermal (long-wave) radiation, air temperature, vapor pressure, wind speed, soil temperature, and net solar (short-wave) radiation. All of these images can be visualized in the 3D environment of the VWP. This allows the user to check for errors, anomalies, and patterns before an actual model run. Using these images as input, ISNOBAL uses a two layer snow model to calculate energy and mass balance terms and produces two images as output. The first is a ten-band energy and mass flux image which includes predicted evaporation, snowmelt, and runoff layers. The second is a nine-band snow conditions image which contains layers for predicted thickness of snowcover, snow density, and mass of the snowcover. Similar to the input images, these output layers can be visualized with the VWP for further analysis. A visualization of several of the ISNOBAL variables can be seen in Figure 2.

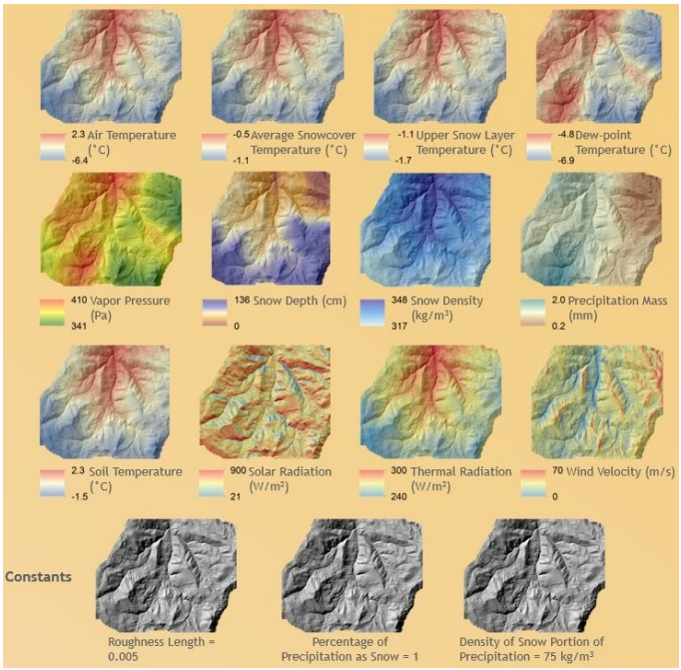


Figure 2: A visualization of different ISNOBAL Variables

3.2 Other Data Models

3.2.1 The DFLOW Model calculates the flows of streams specifically for low flowing streams. DFLOW takes daily stream flow information, and can calculate biologically-based design flows, hydrologically-based design flows, and harmonic and percentile flows. This model can be used to calculate across multiple bodies of water.

3.2.2 The PRMS Model is part of our current implementation. The Precipitation Runoff Modeling System (PRMS) has been developed to simulate the hydraulic processes, simulate the hydraulic water budgets of the watershed, integrate with other models, and have a modular design to allow alternative hydrologic-process algorithms. The hydrologic processes modelled in the system include evaporation, transpiration, runoff, infiltration, and interflow. The interflow is determined by the waterbudgets of plant canopy, snowpack, and soil zones calculated by temperature, precipitation, and solar radiation. Additionally, the hydraulic water budget simulation produces temporal data ranging in scale from days to years [35].

3.2.3 The ParFlow Model (PARallel FLOW) simulates surface and subsurface flow as an integrated hydrological model. This model runs the parallel simulation with an option of three modes. The first two simulation modes, steady-state saturated and variably saturated, show the flow through heterogeneous porous media. The third mode enables the coupling of the surface and subsurface flow. This enables ParFlow to account for the hillslope runoff and channel

routing. ParFlow is the massively parallel computations that run advanced numerical solvers and multigrid preconditioners. This parallel environment takes advantage of octrees to run octree-space partitioning in order to handle input variables in three-space. These features allow ParFlow to run large scale watershed simulations [20].

4 Design and Implementation of the Virtual Watershed

The primary intention for the Virtual Watershed Client is to aid in the visualization of data collected by environmental scientists for better understanding of the watershed environment and the models (ISNOBAL, Parflow, etc.). Several dependencies are needed for the Virtual Watershed Client to be functional. The Virtual Watershed Client requires the Unity game engine, libraries similar to GDAL, and the OGC services. These dependencies in conjunction with the Unity game engine make it possible to implement a geospatial tool and application that is unique in comparison to other applications. The Virtual Watershed Clients architecture, functional requirements, implementation, and main use cases are explained in detail in this section.

4.1 Model View Controller Architecture

The Virtual Watershed Client utilizes a Model-View-Controller (MVC) architectural pattern for the user interface (Figure 3). The Model component is responsible for querying the Virtual Watershed Platform for data, the View component is responsible for displaying the data with the currently applied settings, and the Controller is responsible for handling any configuration changes and movement in the world generated by the VWC. The Model component retrieves data through the OGC services that can be displayed in the form of terrains, shapes (rivers, streams, and roads), model data, and imagery. The model and view component will be discussed more in this section.

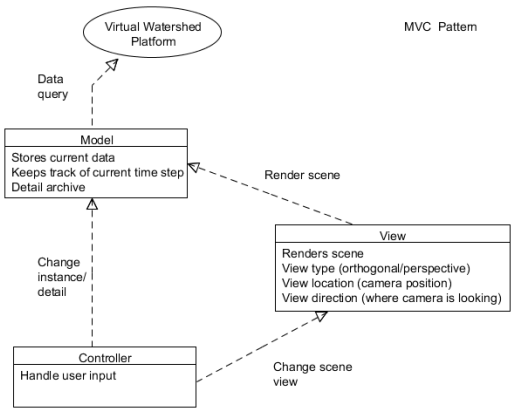


Figure 3: A representation of the Model View Controller architecture of the Virtual Watershed Client

4.2 Use Cases

The users of the Virtual Watershed Client are able to create watershed simulations using data retrieved from the Virtual Watershed Platform. In addition to selecting the data for the simulation, users also have the option of configuring various settings to analyze desired variables of the data in more detail. One of the approaches taken to enable this level of analysis is to allow the user to immerse themselves into the simulation in the sense that he or she can interact with various objects throughout the environment such as trees and sensors. Users also have the option to configure settings for data download speeds, image quality, etc., in order to give greater flexibility in viewing aspects of the data that are most important to the specific users analyses. Once data has been downloaded, users will also be able to configure the representation of that data on the terrain. Figure 4 demonstrates a general use case model [32].

The system being referred to in Figure 4 is the running application of the Virtual Watershed Client [14]. Within the use case model, the server is any outside service that the system utilizes to acquire data. When the application launches, the user can select a dataset to be loaded from a file or downloaded from the server, as well as configure the watershed. The system will use the selected dataset and configurations to start and generate a simulation. Then the Virtual Watershed Client will display the terrain and user/model selected weather conditions. Once the visualization has begun, the user can configure a model run, and have it prepared for simulation by the system. Any data required for the generation of the model run will be acquired from the server at this time. While the model is running, information corresponding to water runoff, snow melt, underground moisture, etc. will be displayed.

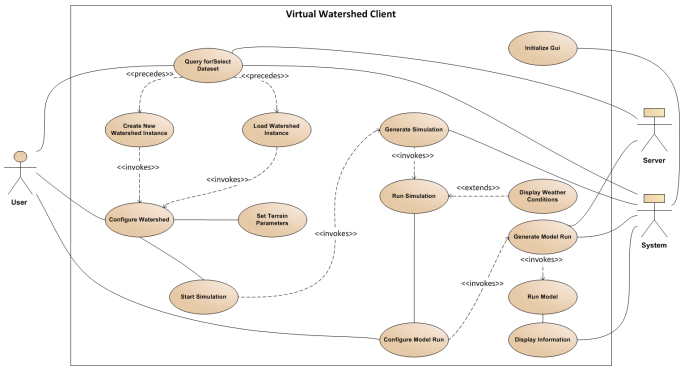


Figure 4: A use case model of the Virtual Watershed Client demonstrates different scenarios from the perspectives of a user, the VWC, and the data server of the VWP

The use cases of Figure 4 are detailed below:

Initialise GUI
The GUI of the Virtual Watershed Client is initialized.

Query for/ Select Dataset
Datasets can be queried for or selected that are acquired from the virtual watershed platform or cached by the VWP.

Create New Watershed Instance
The user has the option of creating a new watershed instance by selecting a different watershed to start in.

Configure Watershed
The VWC can be configured where the user can change what datasets are added to the current watershed.

Start Simulation
Once the watershed has been properly configured the watershed can start loading all necessary datasets and load a scene.

Generate Simulation
Once the simulation is started the scene in Unity will be generated.

Run Simulation
After the scene has been generated, the scene will continue to run automatically.

Set Terrain Parameters
Several options are available for the user to select options such as where the terrain will be located in the real world.

Configure Model Run
The Virtual Watershed Client allows the user to configure a model run for running it.

Generate Model Run
After a model run has been configured it can be generated.

Display Weather Conditions
The virtual watershed has the ability to visualize different weather conditions that the user can select.

Run Model
The user can run models from VWC using the VWP. Currently the VWC has the ability visualize datasets from previously ran models.

Display Information
All of the current datasets, location, elevation, and other useful conditions are provided for the user to view either through the UI or by inspecting the dataset itself.

4.3 Graphical User Interface

4.3.1 The Time Slider shown in Figure 5 on the bottom of the image is a tool that allows users to view data maps over time. Once a model run has been downloaded and selected for viewing by the user, it will be loaded into the time slider, as well as a projector onto the terrain in the data's correct geospatial positioning. There are multiple components of the Time Slider feature. The bar through the center of the Time Slider contains a cursor that shows the current data's position in time related to the overall dataset. The text fields underneath this bar display the information such as the count of the frame currently being displayed, the time that the data corresponds to, how much of the dataset has been downloaded into the Time Slider, and the model run variable that the data represents. On the right side of the Time Slider is a scale configuration tool. This slider allows the user to specify the speed at which they wish to view the data, depending on the overall time differential between the individual data maps. The play button will simply play through the data at the time specified on the speed slider. Finally, the window above the cursor of the Time Slider shows a replica of the data being displayed via the projector. In this way, even if the user is not able to view the data directly on the terrain from their current position, they can see the representation of data played out through time.

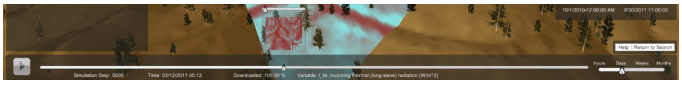


Figure 5: The time slider used to visualize temporal data sets such as model run variables

4.3.2 The Configuration Panel to the left of Figure 6 is used to configure the representation of data in the virtual world. One of the major ways that the data can be configured, is by editing the color-map that is applied to the data. When the data is loaded, the minimum and maximum values are recorded and set to specific color values. All potential values of the data points between the minimum and maximum are then mapped to an interpolated color defined by the five colors of the color-map. These color ranges are predefined at first, but the user can change the value ranges that each color represents. Multiple predefined color-maps are available to the user, with options available to accommodate users with color-blindness. Extending from the color-map is the ability to project the resulting data using either a point or bi-linear interpolation mode. The point interpolation mode simply matches one data point in the grid to a single color value, whereas the bilinear mode blends between the values of the surrounding points to produce a more natural looking representation of the data.

Figure 6: The configuration panel used to alter aspects of data presentation on the terrain

At the bottom of the panel are three text fields, allowing the user to shift the data projection or models in the X, Y, and Z coordinates. This is necessary for cases that a dataset did not come with a predefined position, or that the position given was incorrect. Finally, there are options to export the current data frame being displayed to either an image, showing the color-map representation, or a comma separated value (CSV) file with the values of each point in the data frames grid.

4.3.3 The Data-Point Graph shown in the bottom right of the Figure 7 displays the value at a point on the dataset throughout the time-line of the dataset. The user is able to select a single location on the projected data and the graph will display the data values at that spatial location plotted over the time range of the dataset. The vertical yellow bar of the graph shows the user where in time the currently projected data frame is in relation to the graph as a whole. If the user has two datasets loaded, then there will be two lines present on the graph. This feature allows the user to compare data values of a specific location over two separate model runs or variables. The user will also have the option to download the current graph data as a CSV.

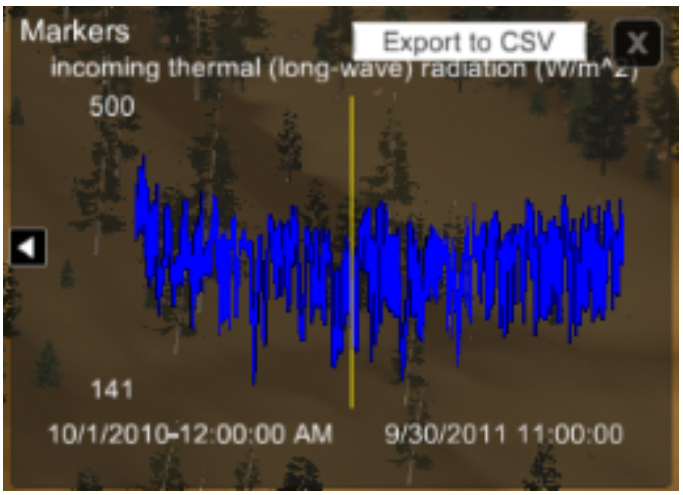


Figure 7: The data-point graph showing the fluctuating data values of a point over time

4.3.4 The Terrain Slicer on the right side of Figure 8 is used for taking 2D cross-sections of the terrain. Two slicer nodes can be placed in the world, and the window of the Terrain Slicer will show a 2D image representing the elevation of the terrain between the two nodes placed. This feature allows for exciting future developments, including the ability to take slices of not only the terrain, but 3D datasets. For example, ParFlow data that models underground water flow could be displayed using the slicer to visualize water flow or ground moisture at differing depths.

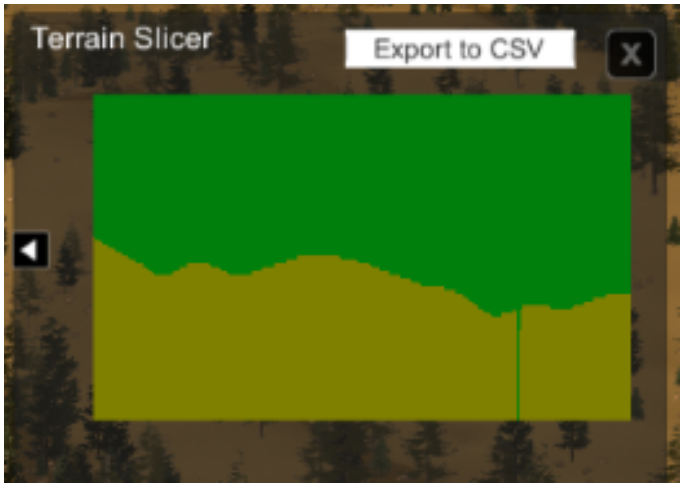


Figure 8: The terrain slicer showing a 2D slice of the environment height-map

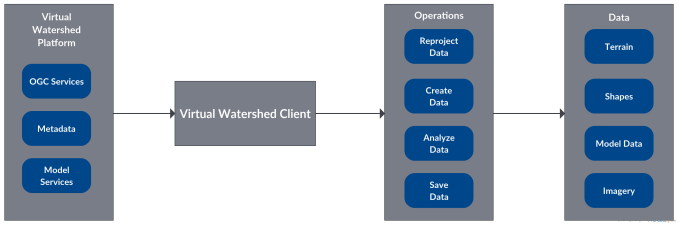


Figure 9: A programmatic flow diagram of the Virtual Watershed Client. The VWP box indicates some of the different services that the VWC can communicate with and the operations box indicates the different operations that the VWC can take on the data. The data box demonstrates the core types of data that the VWC receives from the VWP

4.4 Implementation

4.4.1 Virtual Watershed Client The VWC has several dependencies that make it possible to bring geospatial datasets into Unity for visualization and for analysis. Unity was chosen to be used as the visualization due its cross platform capabilities and its relative ease of use in developing visualizations that could take longer in lower level graphics pipelines such as OpenGL or DirectX. Unity actually handles the lower level graphics pipelines in its own implementation. These dependencies are GDAL, ProjNet, JSON.net, and Gavaghan Geodesy library. GDAL can parse several different geospatial data formats such as geotiffs, netcdf, and many other formats. Both GDAL and ProjNet include the core functionality to do different types of geospatial calculations such as to reproject one coordinate system into another. JSON.net was included in the VWC due to its capability to parse JSON files and has been primarily for parsing the RESTful services coming from the VWP. Figure 9 demonstrates the functionality of the VWC in a block diagram.

The VWC makes use of the RESTful services from the VWP to provide the user with data to generate a watershed and visualize different model runs stored on the VWP. One key thing that the VWC has to do is be able to internalize these datasets into a common form that the application can then generate the proper visualization. Several different parsers have been written using the dependencies described previously to bring these datasets into the VWC. As the datasets are acquired from the Virtual Watershed they will be cached onto local file system to allow for fast loading if the dataset is selected again for loading. These parsers allow us to extract data to build visualizations of many different datasets such as terrains, models that were explained previously, shape files, and images such as landsat imagery. These visualizations are made possible by the API and functionality provided by Unity. Another key feature of the VWC is the ability to incorporate other data sets granted that another parser is written for that particular server.

4.4.2 Data Capabilities The Virtual Watershed Client has the capability to load datasets from the VWP or from the local file system. The VWP serves many different types of data in various formats such as NetCDF [31], TIFF, shapefile, JSON, XML, CSV, and other formats supported by GDAL due to the OGC services supported by the VWP. We use GDAL to parse these formats whether they are from a web service inside of the VWP, or from the local file system into a common data format that is understandable to the VWC. GDAL effectively expands the VWC and allows us to support multiple formats. Along with GDAL we have designed parsers for the VWP that allows us to parse and pull data out of the VWP itself.

The VWC has been designed in such a way that other web services can be easily incorporated as long as the perspective parsers are written for the web service. These web services can be other servers such as NASA Modis's [26] web services. With the incorporation of web services into the VWC, the ability to add other datasets and models is now possible.

With the ability to acquire datasets both locally and the VWP, the VWC is able to create different types of visualizations. These visualizations include procedurally built terrain, bring shape files that may be weather stations or roads, load imagery, and temporal-spatial data sets at run time rather than being preprocessed. With the ability to acquire datasets from web services, it will be possible to create other different types of visualizations such as procedurally vegetation, running models inside the VWP, and uploading datasets.

4.4.3 Coordinate System Design The VWC supports a variety of coordinate systems such that it can represent world datasets in a 3D world and takes into consideration the distance in meters between two objects in order to properly map an object into the virtual world. Our coordinate system utilizes libraries such as GDAL and Gavaghan's Geodesy Library [9] to perform operations such as handling the placement of entities into the world, calculating distances between two points, and changing the projection of other datasets into the VWC's coordinate system. These libraries have allowed us to create a robust coordinate system that can handle a variety of datasets.

In our coordinate system we have decided to use the Universal Transverse Mercator (UTM) coordinate system projections to preserve euclidean relationships between two points. The UTM coordinate system allows the VWC to represent objects with accurate relational distances. Figure 10 demonstrates the UTM coordinate system's 60 longitudinal zones that the Earth is sliced into; measurements using UTM coordinates are usually done within a single zone. A GPS will make use of this coordinate system for relatively short distances due to its accuracy. We utilize this coordinate system for its accuracy in short distances allowing the VWC to place two objects within a reasonable distance of their real world positions, or to project imagery accurately. Though effective when working within an individual zone, the accuracy of the UTM coordinate system diminishes when taking into consideration points that span across two zones.

The limitations of the UTM coordinate system make it difficult to use datasets that span two zones. Calculating the distance between points overlapping a UTM zone boundary results in major distortions. Additionally, calculations of distance overlapping the equator produce similar distortions. These distortions make it necessary for the addition of other coordinate systems into the VWC that deal with these distortions and effectively estimate the distance for two points.

The coordinate system makes use of a frame of reference for placing real world objects into the virtual world of the Unity Game Engine. This frame of reference is paired with the origin in the Unity Game Engine and a location in the real world. This frame of reference allows the VWC to translate real world coordinates into the VWC's coordinate system. Real world points are coordinates having some geospatial relationship to the Earth. These coordinates can be either Latitude and Longitude, UTM points, or other types of coordinates. Unity Game Engine coordinates refer to points that are inside Unity Game Engine with one unit in the Unity Game Engine represented as one meter. This frame of reference makes it easy to convert

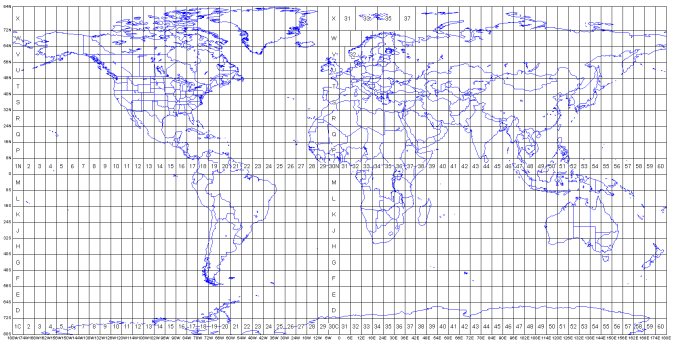


Figure 10: A cartographic projection of the Earth demonstrating the UTM Latitudinal and Longitudinal zones. Typically UTM uses the Longitudinal zones where each has a width of six degrees [25]

between the real world and the VWC's visualization. The frame of reference requires coordinates to be in the Latitude and Longitude system. We utilize Latitude and Longitude points since they are optimal in distance equations, and are easily converted to other coordinate systems. Our coordinate system first converts real world points into a Latitude and Longitude coordinate representation given that they were not originally presented in that format. Algorithm 1 demonstrates how the VWC can convert from real world points to the Unity coordinate system and Algorithm 2 demonstrates the inverse.

The two algorithms utilize the reference coordinate points in order to move between the Earth's coordinate system and Unity's coordinate system. The VWC uses two formula's (Haversine and Vincenty) to calculate the distance between two points. The implementation for Haversine comes from [5], and the implementation for Vincenty comes from [9]. Algorithm 1 is able to handle cases where a data point may be outside the zone currently being observed by the VWC. To account for these cases, Algorithm 3 is used to estimate the equivalent point in Unity. This algorithm estimates a projected point using either the Haversine or Vincenty formula based on the converted Latitude and Longitude calculated in Algorithm 1, and the world Latitude Longitude reference point. Despite the effectiveness of the Haversine and Vincenty formula, they are only accurate to a certain degree. Both of these formula's will work relatively well, however they may not accurately represent the scale of the UTM coordinate system. To account for these inaccuracies our system will allow the user to move datasets if their positions in the virtual world differ from the proper location.

Algorithm 1 World to Unity

```
1: function WORLD TO UNITY(WorldPoint)
2:     ConvertedWorldPoint ← CONVERTTOLATLONG(WorldPoint)
3:     Zone = CALCULATEZONE(ConvertedWorldPoint)
4:     if Zone ≠ WorldZone then
5:         ConvertedWorldPoint ← ESTIMATEPROJECTION(ConvertedWorldPoint)
6:     else
7:         ConvertedWorldPoint ← CONVERTTOUTM(ConvertedWorldPoint)
8:     end if
9:     return (UnityReferenceOrigin+
10:                (UTMWorldReferencePoint−ConvertedWorldPoint))
11: end function
```

Algorithm 2 Unity to World

```
1: function UNITY TO WORLD(UnityPoint)
2:     ConvertedWorldPoint ← CONVERTTOLATLONG(UTMWorldReferencePoint + (UnityReferenceOrigin − UnityPoint), ZoneNumber)
3:     return ConvertedWorldPoint
4: end function
```

Algorithm 3 Estimate Projection

```
1: function ESTIMATEPROJECTION(WorldPoint)
2:     if Vincenty then
3:         Distance ← VINCENTYDISTANCEESTIMATION(WorldPoint, LatLongWorldReferencePoint)
5:     else if Haversine then
6:         Distance ← HAVERSINEDISTANCEESTIMATION(WorldPoint, LatLongWorldReferencePoint)
8:     end if
9:     Direction ← WorldPoint − LatLongWorldReferencePoint
10:    ProjectedWorldPoint ← Distance ∗ Direction
11:    return ProjectedWorldPoint
12: end function
```

5 Virtual Watershed Examples: Lehman Creek and Dry Creek

The Unity engine is a platform for building interactive virtual environments. It provides an integrated development environment (IDE) consisting of a GUI and scripting to define the appearance and behavior of objects in the virtual environment. The Unity engine allows developers to combine 2D and 3D visual assets, audio, and network access to data. The Unity engine is built with Mono, which is a cross-platform open source implementation of the .NET framework [24, 22]. Using Mono allows for procedural control, creation, and manipulation of objects in UnityScript (Javascript) and C#. The engine provides a well optimized code base capable of displaying high-fidelity interactive 3D virtual environments with real-time physics and shading. Although Unity offers great potential to geoscience visualization and education outreach, the IDE has a steep learning curve and is not natively equipped with essential features like geo-referencing, Gregorian time, or the ability to handle geospatial datasets (GeoTiffs, ShapeFiles, etc.) common to GIS Desktop solutions.

The Virtual Watershed Client will provide a simple GUI for communicating to a centralized Virtual Watershed Platform. Acquiring datasets for visualization is now possible through the VWP within the VWC or the local file system. Several visualizations have been developed for the Virtual Watershed Client using preprocessed files from the local file system. Datasets acquired from the VWP can be used to procedurally build a terrain for visualization at run time of the Virtual Watershed Client. We have constructed preprocessed terrains for the WC-WAVE project that are being used for analyzing datasets that are specific to the WC-WAVE project.

The process of building preprocessed terrains requires certain datasets to create a well built visualization. When available, high-resolution LiDAR can be represented by the

terrain. High-resolution orthoimagery is then placed on the terrain, and procedurally splatted with detailed textures and vegetation (as shown in Figure 14 and Figure 16). Both Figure 14 and Figure 16 can be compared to their respective counterparts, Figure 13 and Figure 15, that demonstrate the overall effectiveness of our visualization.

Once a base map has been constructed, scientific data can be incorporated. A projector with a customized shader allows for choropleths (thematic maps) to be overlaid onto the terrain as shown in Figures 11 and 12. These maps could represent characteristics of terrain topology like slope or spatial-temporal variables associated with hydrology models such as ISNOBAL, Parflow, or PRMS. An example of a spatial-temporal output of an ISNOBAL variable can be seen in Figure 11.

We can also visualize the outputs of a PRMS model by playing back the output file onto the terrain as a choropleth. As explained previously in Section 4, the user can alter the values of the projector, change the times on the time slider, or compare two model runs.

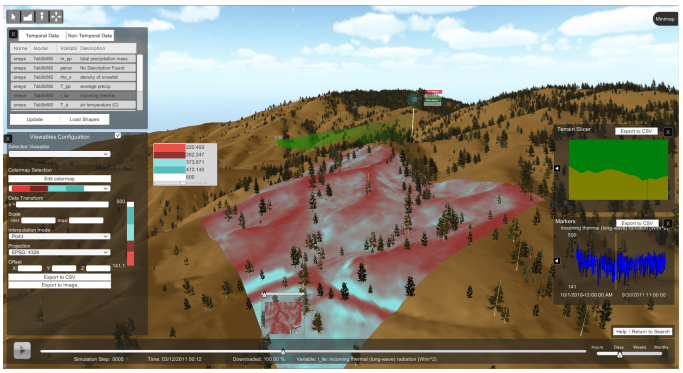


Figure 11: The entire user interface shown on the Dry Creek terrain with a thermal long wave radiation dataset projected on to the surface

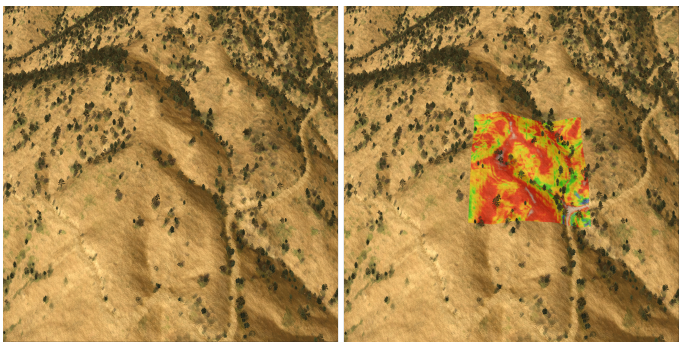


Figure 12: A picture of the Dry Creek sub-catchment with choropleth (right) and without choropleth (left)

6 Future Work and Conclusions

We have covered the design and implementation of the Virtual Watershed Client, the different models of the WC-WAVE project, and new additions to the VWC. This paper discussed the unique design of a three-dimensional project for the use with geospatial data. The project incorporates the GDAL libraries into a game development engine, Unity3D, for purposes of research. Additionally the unique tools designed into the project to interact with the data will help researchers gather more information from the provided data. The VWC has several new features that include elements that have been added to the UI, a new coordinate system, the ability to visualize spatial-temporal datasets, generate terrains procedurally, and the incorporation of the Virtual Watershed Platform. With the addition of these new features researchers can visit different areas in the world and utilize the new features for the purposes of analyzing datasets that are both stored locally and on the VWP.

In terms of future work the team is planning new features that would be both beneficial and needed for the Virtual Watershed Client. Integration with the VWP has allowed us to incorporate procedural terrain generation and allows us to consider implementing procedural vegetation generation. Being able to procedurally generate vegetation alongside terrain would allow us to simulate different models such as CASiMiR that have a vegetation output. Watershed researchers could effectively analyze the effect of changing seasons, climate, and fire scenarios. We will soon incorporate the ability to visualize vector motion fields that would be useful for showing stream flow in the bottom of a river or to demonstrate wind. To provide a more immersive experience of the Virtual Watershed Client with virtual reality, we would like to allow the user to to use Head Mounted Displays (HMD) such as the Oculus Rift [27], HTC Vive [12], or Google Cardboard [11]. We would like to also build a visualization for a six-sided CAVE (Cave auto virtual environment). We would like to incorporate other sensors such as the Microsoft Kinect [21] and others. Beyond new visualizations features and the incorporation of other sensors we have plans to incorporate other services from the Virtual Watershed Platform and other servers.

The Virtual Watershed Client currently is able to interface with the Virtual Watershed Platform and download datasets from it. We would like to expand our functionality to allow users to upload datasets to the Virtual Watershed Platform and run models on those datasets. While the Virtual Watershed Platform support for many other different types of datasets relate to the WC-WAVE project, we would want to include other servers such as NASA's Modis web services [26] that have global datasets that could be beneficial to the Virtual Watershed Client. Incorporation of these other web services would be both beneficial and useful for the Virtual Watershed Client and would allow for a greater diversity of datasets to be visualized and analyzed within the VWC.

Figure 13: A photograph of Dry Creek

Figure 14: The VWC image of Dry Creek

Figure 15: A photograph of Lehman Creek

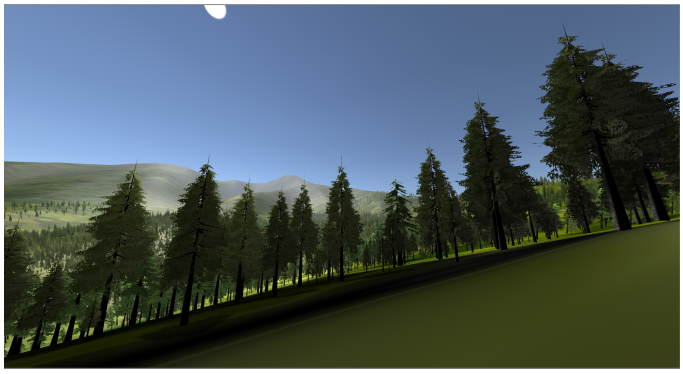


Figure 16: The VWC image of Lehman Creek

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under grant numbers: IIA-1329469, IIA-1329470, IIA-1329513, and IIA-1301726, and the EPSCoR MILES grant number: IIA-1301792. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation

References

[1] V. Alarcon and C. O'Hara, "Advanced Techniques for Watershed Visualization," *Applied Image Pattern Recognition Workshop*, vol. 35, 2006.

[2] E. A. Anderson, "National Weather Service River Forecast System Snow Accumulation and Ablation Model: NOAA Tech," *Memorandum NWS Hydro-17, US Dept. of Commerce*, pp. 3–7, 1973.

[3] B. Berger, R. Hansen, and I. Cowley, "Developing a Virtual Watershed: Sevier River Basin," Decision Support Systems for Water Resources Management (Specialty Conference). American Water Resources Association, 2001.

[4] C. D. Carthen, T. J. Rushton, C. M. Johnson, A. Hesson, D. Nielson, B. Worrell, D. M. Delparte, W. J. Johansen, J. W. Anderson, R. Lew, N. R. Wood, M. Ziegler, S. M. Dascalu, and F. C. Harris, "Design of a Virtual Watershed Client for the WC-WAVE Project," IEEE International Conference on Collaboration Technologies and Systems (CSA), June 2015, pp. 90–96.

[5] D. Dennehy, "Haversine Algorithm in C#," [Accessed on 15 March 2016]. [Online]. Available: http://damien.dennehy.me/blog/2011/01/15/haversine-algorithm-in-csharp/

[6] Esri, "Arcgis Platform," [Accessed on 14 March 2016]. [Online]. Available: www.esri.com/software/arcgis

[7] G. N. Flerchinger and K. E. Saxton, "Simultaneous Heat and Water Model of a Freezing Snow-Residue-Soil System I. Theory and Development," *Transactions of the ASAE*, vol. 32, no. 2, pp. 565–0571, 1989.

[8] K. J. Franz, T. S. Hogue, and S. Sorooshian, "Operational Snow Modeling: Addressing the Challenges of an Energy Balance Model for National Weather Service Forecasts," *Journal of Hydrology*, vol. 360, no. 1, pp. 48–66, 2008.

[9] M. Gavaghan, "C# Geodesy Library for GPS Vincenty's Formula," [Accessed on 1 March 2016].

[Online]. Available: http://www.gavaghan.org/blog/free-source-code/geodesy-library-vincentys-formula/
[10] Gdal, "Gdal - Geospatial Data Abstraction Library," [Accessed on 14 March 2016]. [Online]. Available: http://www.gdal.org/
[11] Google, "Google Cardboard," [Accessed on 29 February 2016]. [Online]. Available: www.vr.google.com/cardboard/
[12] HTC, "Htc vive," [Accessed on 29 February 2016]. [Online]. Available: www.htcvive.com/us/
[13] R. Jordan, "A One-Dimensional Temperature Model for a Snow Cover: Technical Documentation for SNTHERM. 89." DTIC Document, Tech. Rep., 1991.
[14] S. Lauesen, "Task Descriptions as Functional Requirements," *IEEE Software*, vol. 20, no. 2, pp. 58–65, 2003.
[15] G. H. Leavesley, R. W. Lichty, B. M. Thoutman, and L. G. Saindon, *Precipitation-Runoff Modeling System: User's Manual*. US Geological Survey Colorado, CO, 1983.
[16] G. E. Liston and K. Elder, "A Distributed Snow-Evolution Modeling System (SnowModel)," *Journal of Hydrometeorology*, vol. 7, no. 6, pp. 1259–1276, 2006.
[17] D. Marks, J. Domingo, and J. Frew, "Software Tools for Hydro-Climatic Modeling and Analysis: Image Processing Workbench, ARS-USGS Version 2," *ARS Tech. Bull. 99*, vol. 1, 1999.
[18] D. Marks, J. Domingo, D. Susong, T. Link, and D. Garen, "A Spatially Distributed Energy Balance Snowmelt Model for Application in Mountain Basins," *Hydrological Processes*, vol. 13, no. 12-13, pp. 1935–1959, 1999.
[19] ——, "A Spatially Distributed Energy Balance Snowmelt Model for Application in Mountain Basins," *Hydrological Processes*, vol. 13, no. 12-13, pp. 1935–1959, 1999.
[20] R. M. Maxwell, S. J. Kollet, S. G. Smith, C. S. Woodward, R. D. Falgout, I. M. Ferguson, C. Baldwin, W. J. Bosl, R. Hornung, and S. Ashby, "ParFlow User's Manual," *International Ground Water Modeling Center Report GWMI*, vol. 1, no. 2009, p. 129, 2009.
[21] Microsoft, "Kinect for Windows," [Accessed on 2 March 2016]. [Online]. Available: http://www.microsoft.com/en-us/kinectforwindows/
[22] ——, ".NET Framework and .NET SDK Downloads," [Accessed on 3 March 2016]. [Online]. Available: https://msdn.microsoft.com/en-us/vstudio/aa496123.aspx
[23] S. Miller, D. Semmens, D. Goodrich, M. Hernandez, R. Miller, W. Kepner, and D. Guertin, "The Automated Geospatial Watershed Assessment Tool," *Environmental Modelling & Software*, vol. 22, no. 3, pp. 365–377, 2007.
[24] Mono-Project, "Home — mono," [Accessed on 28 February 2016]. [Online]. Available: http://www.mono-project.com/
[25] A. Morton, "Dmap: UTM Grid Zones of the World," [Accessed on 1 March 2016]. [Online]. Available: http://www.dmap.co.uk/utmworld.htm
[26] National Aeronautics and Space Administration, "Modis Web," [Accessed on 3 March 2016]. [Online]. Available: http://modis.gsfc.nasa.gov
[27] Oculus, "Oculus Rift - Virtual Reality Headset for Immersive 3D Gaming," [Accessed on 29 February 2016]. [Online]. Available: www.oculus.com/
[28] Opengeospatial, "Open Geospatial Consortium — OGC," [Accessed on 14 March 2016]. [Online]. Available: http://www.opengeospatial.org/
[29] D. Patel, M. Dholakia, N. Naresh, and P. Srivastava, "Water Harvesting Structure Positioning by Using Geo-Visualization Concept and Prioritization of Mini-Watersheds Through Morphometric Analysis in the Lower Tapi Basin," *J Indian Soc Remote Sens*, vol. 40, no. 2, pp. 299–312, 2011.
[30] QGIS, "Qgis Project," [Accessed on 14 March 2016]. [Online]. Available: http://www2.qgis.org/en/site/
[31] R. Rew and G. Davis, "NetCDF: An Interface for Scientific Data Access," *Computer Graphics and Applications*, vol. 10, no. 4, pp. 76–82, 1990.
[32] I. Sommerville, *Software Engineering*. Boston: Pearson, 2011.
[33] D. G. Tarboton and C. H. Luce, *Utah Energy Balance Snow Accumulation and Melt Model (UEB)*. Citeseer, 1996.
[34] United States Environmental Protection Agency, "DFLOW — Water Data and Tools," [Accessed on 2 March 2016]. [Online]. Available: http://www.epa.gov/waterdata/dflow
[35] United States Geological Survey, "Prms," [Accessed on 11 March 2016]. [Online]. Available: ftp://brrftp.cr.usgs.gov/pub/mows/software/prms/release_notes.pdf
[36] Unity3D, "Unity - Game Engine," [Accessed on 14 March 2016]. [Online]. Available: http://unity3d.com/
[37] USDA Agricultural Research Service, "Ipw Command: isnobal," [Accessed on 14 March 2016]. [Online]. Available: ftp://ftp.nwrc.ars.usda.gov/ipw/Marks%20et%20al%201999/man1/isnobal.html
[38] Western Consortium, "Western Tri-State Consortium," [Accessed on 14 March 2016]. [Online]. Available: http://westernconsortium.org/

Chase Carthen graduated from the University of Nevada, Reno with both a B.S. and a M.S. in Computer Science and Engineering in 2014 and 2016 respectively. He is currently working in industry as a software engineer. His research interests include human-computer interaction, graphics and simulations, and artificial intelligence.

Thomas J. Rushton graduated with a B.S. in Computer Science and Engineering 2016 from the University of Nevada, Reno. His research interests include virtual reality, data visualization, and artificial intelligence. He is currently working in industry as a software engineer.

Nolan P. Burfield graduate with a B.S. in Computer Science and Engineering in 2015 and is working towards a M.S. in Computer Science and Engineering at the University of Nevada, Reno. He works as a research assistant in the High Performance Computation and Visualization Lab. His research interests are in the areas of computer graphics and financial data computation.

Christine M. Johnson graduated from the University of Nevada, Reno with both a Bachelors of Science and a Masters of Science in Computer Science and Engineering in 2014 and 2015 respectively. She is currently working in industry for Intentional Software Corporation as a software engineer.

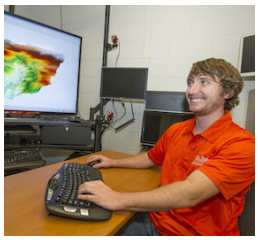
Aaron Hesson graduated with a B.S. degree in Computer Science and Engineering at the University of Nevada, Reno in 2014. His specialization is Intelligent Systems.

Daniel Nielson graduated with a B.S. in Computer Science and Engineering from the University of Nevada,Reno in 2014. He also has a degree in English from the University of Southern California. He is currently working in industry as a software engineer.

Bryan Worrell graduated with a B.S.in Computer Science and Engineering at the University of Nevada, Reno in 2014. His interest lies in the field of Games and Simulations and he is currently working as a software engineer in industry.

Donna Delparte is an Assistant Professor in the Department of Geosciences at Idaho State University. She received her BS in Geography (1989) at the University of Regina, Canada and MS (1998) and PhD in Geography (2008) at the University of Calgary, Canada. Dr. Delparte has an extensive background in the applications of GIS and remote sensing to the fields of geosciences, resource management and conservation/environmental planning. She has active research in using Unmanned Aircraft Systems for conservation mapping and precision agriculture applications and analysis.

Tucker Chapman received his B.S. degree in Geology from Brigham Young University, Provo, UT, USA, in 2015. He is currently pursuing a M.S. degree in Geographic Information Science, being advised by Donna Delparte, at Idaho State University, Pocatello, ID, USA and is expecting to graduate in 2017. He is the lead developer of the gridding tool, including performance testing, its REST and Python APIs, and its ArcGIS package.



W. Joel Johansen completed his Masters in GIS in 2015 at Idaho State University and is currently working as a Precision Agriculture Specialist for the J. R. Simplot company. He received a BS in Geology from Brigham Young University in 2013. His current job includes database management, data processing, software development, and analyzing satellite imagery to monitor crop health.

Roger Lew has a B.S. in psychology from University of Idaho in 2004, a M.S. in human factors psychology from University of Idaho in 2007, and a Ph.D. in neuroscience from University of Idaho in 2014. He is currently working as a Research Assistant Professor to manage the Virtual Technology Laboratory and conduct research related to social-ecological systems and nuclear human factors.

Nicholas R. Wood is currently working on his Masters of Science in Integrated Architecture and Design, at the University of Idaho. He received his Bachelors of Science in Virtual Technology and Design at the University of Idaho in 2014. His research interests are integration of data based visualization and Video game narrative.

Matthew Ziegler received his Bachelors of Science in Virtual Technology and Design at the University of Idaho in 2014.

John W. Anderson is an Associate Professor in the College of Art and Architecture, co-founder of the Virtual Technology & Design program and Co-Director of the Virtual Technology Laboratory at the University of Idaho. His teaching and research are focused in the areas of trans-architectures, tele-present environments, evolutionary game theory and design visualization technologies with an emphasis on complex system design and analysis. He is a design thinker who leads interdisciplinary communities of virtual design experts, scientists, engineers, educators, and artists where the focus is the incorporation of virtual technology in all aspects of education, research, modeling, and simulation.

Sergiu M. Dascalu is a Professor in the Department of Computer Science and Engineering at the University of Nevada, Reno, USA, which he joined in 2002. In 1982, he received a Master's degree in Automatic Control and Computers from the Polytechnic University of Bucharest, Romania and in 2001, a Ph.D. in Computer Science from Dalhousie University, Halifax, NS, Canada. His main research interests are in the areas of software engineering and humancomputer interaction. He has published over 140 peerreviewed papers and has been involved in numerous projects funded by industrial companies as well as federal agencies such as NSF, NASA, and ONR.

Frederick C. Harris Jr. is currently a Professor in the Department of Computer Science and Engineering and the Director of the High Performance Computation and Visualization Lab and the Brain Computation Lab at the University of Nevada, Reno, USA. He received his BS and MS in Mathematics and Educational Administration from Bob Jones University in 1986 and 1988 respectively, his MS and Ph.D. in Computer Science from Clemson University in 1991 and 1994 respectively. He is a member of ACM (Senior Member), IEEE, and ISCA (Senior Member). His research interests are in parallel computation, computational neuroscience, computer graphics and virtual reality.