https://doi.org/10.2352/ISSN.2470-1173.2017.5.SD&A-374

Stereo rendering of photorealistic precipitation

Syed Hussain, David F. McAllister; Department of Electrical and Computer Engineering, North Carolina State University; Raleigh, NC

Abstract

The stereoscopic rendering of rain has been previously studied. We extend the behavior and distribution of rainfall to include photorealistic stereo rendering of rain and snow precipitation at video frame rates. We ignore stereo rendering optimization and concentrate on the visual factors necessary to produce photorealistic output. The experimental method uses a series of controlled human experiments where participants are presented with video clips and still photos of real precipitation. The stimuli vary along three visual factors: particle numbers, particle size, and motion. The goal is to determine the statistical ranking and importance of these visual factors for producing a photorealistic output. The experiments are extended to investigate if stereo improves photorealism. Additionally, experimental stimuli include post-processing on rendered output to produce variable lighting, glow, and fog effects to study their impact on photorealism as the stereo camera moves in the scene. The results demonstrate that the visual factors for photorealism can be ranked as more sensitive to particle numbers and motion than to particle size. Varying light, glow, and fog effects contribute towards photorealism independent of stereo. Future research will exploit the geometric symmetry of the stereoscopic image pairs to render precipitation while maintaining realtime frame rates.

Introduction

It is important to understand formation of precipitation in nature before producing realistic computer generated rendition of such natural phenomenon. In nature, water vapors are formed by the process of evaporation. They are carried upwards by air drafts produced by variations in atmospheric temperature. At a higher altitude, cooler temperatures cause water vapors to condense around airborne particles to form clouds. As the weight and size of these particles increase beyond a certain threshold of keeping them afloat, they fall towards the ground in form of either liquid, solid or mixed precipitation such as rain, snow, hail, freezing rain, or drizzle. This cycle of surface moisture to convert into water vapors and come down in various forms of precipitation is called the *water cycle*, also known as the *hydrologic cycle* [1]. Although natural precipitation exists in many forms, in this study computer generation of rain and snow is considered.

There are several applications of synthetic precipitation phenomenon found in the movies, video games, and virtual reality. Special effects are often used in the movies but they require expensive and time consuming off-line processing to achieve photorealism. On the other hand, video games require a balance between realtime user input and photorealistic output. Evolution of software algorithms and recent improvements in computer hardware have made it possible to produce realtime frame rates with emphasis on photorealism. Introduction of stereo rendering of natural phenomena in a scene provides visual realism and immersive feel relevant to many true 3D computer graphics and virtual reality applications. It is a challenging problem due to the complex interaction of rain or snow particles with light and the dynamic nature of precipitation such as variations in number, size, and motion of particles. However, introduction of natural phenomena greatly enhances a scene and gaming experience. A comprehensive survey of recent work in data acquisition and computer simulation of natural phenomena is done by Zhao [2].

Our goal is to produce photorealistic computer generated stereo output at video frame rates of at least 120 frames per second (fps). Photorealism is defined as a detailed representation like that obtained in a photograph in a non-photographic medium such as a painting or, in our case, computer graphics. We use an experimental method for measuring the perceived visual realism and show the results of a series of controlled human experiments which present factors important to achieving a photorealistic precipitation scene. We extend our previous study of rendering rain in stereo to include investigation of photorealistic stereo output for rain and snow [3].

Literature Review

References to photorealism appear in several studies. However, characterization of the term varies. This is because human response to visual stimuli is as variable as individuals themselves. Rademacher et al. [4] defines photorealism as an image that is perceptually indistinguishable from a photograph of a real scene. Since a photograph is planar, this characterization of photorealism is well suited for monoscopic view. It is human consciousness that identifies a scene as real or otherwise. Our brain creates a perception of reality after processing visual information received by our two eyes, which are taking a snapshot of the world around us from two different perspectives. A stereoscopic characterization of photorealism is closer to visual realism where depth is perceived by presenting images from two different perspectives.

In a previous study, we presented a method for stereo rendering of rain in realtime [3]. However, we do not consider complex illumination and therefore produce a realtime stereo output without emphasis on photorealism. Stereoscopic realtime output in other natural phenomena, such as fire [5], vegetation [6], fog, mist, and clouds [7] have been studied. Rose and McAllister [5] illustrate a realtime photorealistic stereoscopic rendering method to depict fire. The authors use several layers of high-quality pictures of fire as textures to attain photorealistic effects. This works well for fire but natural phenomena, such as rain or snow, is distributed over larger 3D space and therefore representing each particle with layers of static images is not feasible to produce visually realistic animation of rain or snow particles. Pre-rendered high-quality images or actual photographs used to produce photorealistic computer generated images is referred to as image-based rendering. A survey of several image-based rendering methods addressing different rendering problems have been studied by Zhang and Chen [8].

Another way to achieve photorealistic results is to use actual photographs or a video sequence to show the background while superimposing synthetic precipitation in the foreground. The lighting parameters are pre-computed from the background image and applied to the rendered particles. A method to overlay synthetic rain on real video sequences by studying visual properties of rainfall in terms of time and space is described by Starik and Werman [9]. The authors assumed partial knowledge of intrinsic camera

parameters, such as focal length and camera exposure time, and user-defined rain parameters, such as intensity and velocity, to achieve photorealistic results. A similar study, Mizukami et al. [10] propose a method to render a realistic rain scene that represented environmental conditions that change from one scene to another. They modeled wind effect, intensity, and density of rainfall. They also proposed a technique to calculate raindrop trajectory that made rainfall appear more realistic.

In another study, Garg and Nayar [11] illustrate photorealistic rendering of rain with a system that measures rain streaks. A rain steak is formed when a raindrop looks like a bright stroke on an image due to its fast movement relative to the camera exposure. In humans, falling raindrops are perceived as streaks due to retinal persistence. The authors presented a model for rain streak appearance that captures these complex interactions between various lighting directions, viewing directions and oscillating raindrop shapes. To fully capture all the possible illumination conditions, they constructed an apparatus to photograph distortion in a falling raindrop from various angles. From these experiments a large database consisting of thousands of rain streak images is created that capture variations and appearance of a falling raindrops with respect to light position and view direction. Subsequently, an image-based rendering algorithm is applied to add rain to a single image or video.

A video of a natural rain scene is used by Wang et al. [12] to apply an off-line image analysis method to create a rain mask, which is extraction of rain streaks from that video. Several rain streaks are selected from the rain mask and used in a particle system to render rain. Precomputed environment lighting of the scene is mapped to the synthetic rain. To make the rain part of the scene it is superimposed on the scene. The rain properties such as intensity and speed can be changed in realtime. The implementation takes advantage of graphics hardware. Photorealism is achieved by applying environment lighting and image-based rendering.

Existing monoscopic rendering of rain is extended by Creus and Patow [13] that add splashes and illumination effects, such as fog, halos, and light glows using image-based rendering. Their algorithm does not impose any restriction on rain area dimensions but instead only render rain streaks in the area around the viewer. The simulation runs entirely on the graphics processing unit (GPU), which includes collision detection with other scene objects and use precomputed images for illumination effects. In another study by Wang et al. [14], an image-based rain scene rendering is extended to include cloud lightening and wind effects that enhances realism. Photorealistic images are generated by implementing a ray-tracing algorithm to render rain under different lighting conditions. The authors also simulate hazy atomization to create mist simulation by fusing raindrops texture into the scene background. Rain dynamics are implemented using a particle-based system.

The methodologies used to simulate and render rainfall can also be applied to snowfall such as scrolling texture-based or particle-based systems. A scrolling textures method is used by Wang and Wade [15] for both rain and snow. The authors applied snow textures to a double cone that was placed around the observer. Applications such as flight simulation can take advantage of such a method. Although it can be implemented in realtime, it lacks realism due to little or no interaction with other objects in the scene. On the other hand, the particle-based systems are more physically realistic since each snowflake is simulated and rendered individually whose motion is determined by the influences of various forces, such as gravity, wind, and air resistance. Another method to render rain and snow is proposed by Yang et al. [16] in which the authors combined two techniques, namely *level-of-detail*, which decreases complexity of a 3D object representation as it moves away from the viewer, and *fuzzy-motion*, the blurriness of moving objects due to persistence of vision. Such combinations are used to increase particle system efficiency for rendering of natural phenomenon. Instead of using a billboard method to represent rain or snow, the particles are expanded to form a *point eidolon* that can be texture mapped. A point eidolon is defined as a point that is stretched to conform to a rain streak or snowflake shape, which is texture mapped with appropriate rain or snow texture. The authors show that the number of polygons required to create a point eidolon are lower than creating a billboard, increasing the performance of the particle system.

The realism of snow can also be enhanced by having snow accumulate on the ground and on other scene objects. A method by Fearing [17] is presented for generating snow cover, which consist of two models that address snow accumulation and stability. The accumulation model calculate how much snow each surface should receive. This is based on a counter-intuitive idea where particles were emitted from upward-facing surfaces towards the sky to determine exposure to falling snow. The amount of exposure determine the amount of snow accumulation on the surface. To simulate wind effects, the author used a global wind direction that had an advantage of not requiring any fluid computations. However, it did not produce fully convincing accumulation patterns. The stability model is used when layers of snow are added to the scene in an unstable area to determine if it will fall downward. The method is based on calculating the *angle of repose*, which was used to measure the static friction between piles of granular material. If the repose angle is too steep, snow is redistributed. To render the scene, the author used commercial rendering software. While the method is visually superior and photorealistic, it is computationally expensive. This work is extended by Feldman and O'Brien [18] who present a new method for accumulating snow with realistic wind effects. They applied fluid dynamics techniques described by Fedkiw et al. [19] to create wind velocity fields and use it to drift accumulated snow in a more realistic manner. The snow is accumulated by storing the amount on a horizontal surface of a three-dimensional grid, a voxel, which is marked as occupied. New voxels is marked as occupied after enough snow had been accumulated to fill the voxels beneath.

Photorealistic heavy snowfall requires use of a large number of particles. A frequency domain spectral analysis of the rendered image to reduce the number of particles in snowfall, yet keep it visually realistic, is described by Zou et al. [20]. They combine geometry-based falling particles with image-based spectral synthesis. The method first render an image with simple particle-based snowfall representation. It then analyze the movement of particles stored inside an image within the frequency domain. This is used to produce an opacity map, which creates an illusion of denser snowfall.

An efficient snow distribution method is presented by Festenberg and Gumhold [21] to give a realistic snow cover on object surfaces, as an alternative to using high-cost particle system simulation, which produced simplified snow surface representation. The author use photographs of snow-covered scenes as a primary source for the model development. Inspired by real world observations, they derive a statistical snow deposition model.

Visual Factors for Photorealism

Physically accurate environment lighting is another approach to produce visually realistic results. However, such physically-based rendering alone is not sufficient to achieve photorealism. There are

many visual factors that contribute to generate photorealistic results. Rademacher et al. [4] demonstrate that sharp edges make objects look artificial. Making sharp edges smooth and slightly rounded gives an object model a more realistic look. Similar conclusions are drawn for shadows. A sharp contrast of a dark shadow looks artificial while soft shadows make a computer-generated scene look realistic. Imperfections in camera lens can produce distortions such as lens flare, chromatic distortion, or the formation of out-of-focus areas. Reproduction of these imperfections in a computer-generated image adds to photorealism although we do not consider these distortions in this research.

A fundamental challenge associated with achieving photorealism in a rain or snow scene is lighting calculations. In previous studies involving particle systems, Garg and Nayar [11] investigate photorealistic rain rendering with lighting effects. The study involves interaction of light with oscillating raindrops which produce complex brightness patterns such as speckles and smeared highlights. The importance of light attenuation to achieve photorealism is demonstrated by Tatarchuk and Isidoro [22]. Atmospheric effects such as fog and addition of glow effects around light sources such as misty halos around streetlights for nighttime rendering is also considered.

Precipitation scenes consist of several thousands of particles that move independently at variable velocities under the influence of external forces such as gravity, air resistance, and wind gusts. The intensity of rain or snowfall is dependent on the number of particles and particle size. A light precipitation event will have fewer and smaller particles as opposed to heavy precipitation. Similarly, the variability in a particle shape is due to external forces acting on the falling particles. Variations in particle details due to changes in shape, size, number of particles, and observational distances are important visual factors. The experiments proposed in this study collect subjective data that is analyzed to rank three factors important for photorealism of a computer-generated rain or snow scene.

Particle Shape and Motion

The changing shape of a falling raindrop is a result of a constant tug-of-war between surface tension of the raindrop and the pressure of the air pushing up against the bottom of the drop. When the drop is smaller than 2 mm, surface tension wins and pulls the drop into a spherical shape. The fall velocity increases as a raindrop gets bigger which causes the pressure on the bottom to increase. The raindrop becomes more oblate with a flatter surface facing the oncoming airflow. Larger drops become increasingly flattened at the bottom forming a depression. In raindrops that are greater than 4 mm across, this depression grows to form a parachute shape that explodes into many smaller droplets. Examples of this phenomenon are described by Pruppacher and Klett [23].

The shape and motion of a snowflake changes as it collides with other snowflakes and melt together. The shape of individual snow crystals are classified into eight primary categories [24]. The shape is also influenced by the temperature and humidity of the atmosphere. Snowflakes form in the atmosphere when water droplets freeze onto dust particles. Depending on the temperature and humidity of the air where the snowflakes form, the resulting ice crystals will grow into a myriad of different shapes.

Particle Size

In general, a raindrop size can vary from 0.1 mm to 9 mm across. The size of a raindrop is typically greater than 0.5 mm in diameter. In widely scattered rain, the drops may be smaller than 0.1 mm. However, often droplets larger than about 4 mm can become unstable and split into smaller droplets. A study by Srivastava [25] shows that the probability of a raindrop breaking up is given as a function of its size and results in an exponential increase in the likelihood of breakup in droplet sizes beyond 3.5 mm.

The size of a snowflake varies greatly. The smallest snowflakes are called diamond dust crystals. They are as small as 0.1 mm. These faceted crystals sparkle in sunlight as they float through the air, which is how they got their name. They form rarely in extremely cold weather. The largest snow crystal ever recorded measured 10 mm from tip to tip.

Number of Particles

Variation in the number of raindrops impacts rain intensity, which is measured by a volume of water accumulated per unit of time. The more raindrops, and therefore water, the greater the rain intensity. This is commonly classified into three categories: light, moderate and heavy rain. The rate of light rain varies between a trace and 2.5 mm/hr (millimeters per hour). Moderate rain rate is between 2.5 to 7.5 mm/hr. Rain is classified as heavy if the rate is more than 7.5 mm/hr.

Like rain, snow is classified into light, moderate and heavy categories. Snow intensity is measured by looking at how much equivalent liquid water a snowfall generates. This is called a *liquid water equivalent measurement*. Light snow rate is about 1.0 mm/hr, moderate rate is between 1.0 to 2.5 mm/hr, and it is classified as heavy if the rate is more than 2.5 mm/hr.

Implementation

Particle System

A particle system, which is first introduced by Reeves [26], is a technique well suited to simulate and render particles that do not have smooth or well-defined surfaces but instead are irregular in shape and complex in behavior. Since rain and snow consist of thousands of tiny particles, a particle system is well suited to implement such natural phenomenon. A particle system is a large set of simple primitive objects, such as a point or a triangle, which are processed as a group. Each particle has its own attributes including position, velocity, and lifespan that can be changed dynamically. Particles move and change their attributes over time before their extinction, which occurs when the particle lifespan is exhausted or where an attribute falls below a specified threshold. If the attributes of particles are coordinated, the collection of these particles can represent an object, such as rain or snow. To achieve the desired effects of rain or snowfall, many independent particles are simulated and rendered simultaneously. Pre-computed images of rain streaks and snowflakes are used as billboards to render an appropriate precipitation scene. This increases rendering speed and enables the same particle system to simulate other phenomena such as falling autumn tree leaves by changing the billboard texture and modifying particle attributes, such as particle mass and velocity.

Realtime

In the particle system implemented on a GPU, each particle cycles through the two key program stages running on a GPU. One is responsible for particle simulation and the other is required for particle rendering in stereo. The desired video frame rate is attained by taking advantage of inherent parallel architecture of a modern GPU, which can operate in *shader* or *compute* mode. The name shader is a misnomer since it has little to do with various lighting or shading models. A shader is an executable program that runs on a GPU and processes data in parallel. In the Open Graphics Library (OpenGL), it is written in a C-like language called the OpenGL

Shading Language (GLSL). The modern graphics pipeline is defined by various programmable shader stages namely, vertex, tessellation, geometry, and fragment shaders. Each stage processes incoming data for the next stage. In the shader mode, required vertex shader and fragment shader programs, along with optional geometry and tessellation shader programs, form a modern OpenGL graphics pipeline.

In compute mode, the compute shader is a new programming stage added to the modern OpenGL version 4.3. This enables a GPU to be used for general purpose computing. The particle simulation for rain and snow is performed in GPU compute mode using a compute shader for better OpenGL interoperability. It is also used to simulate environment effects, such as gravity and wind to render a particle system. The compute shader uses two buffers; one stores the current velocity of each particle and a second stores the current position. At each time step, a compute shader updates position and velocity values. Each invocation of a compute shader processes a single particle. The current velocity and position are read from their respective buffers. A new velocity is calculated for the particle and then this velocity is used to update the particle's position. The new velocity and position are then written back into the buffers. To make the buffers accessible to the compute shader program, a *shader storage buffers object* (SSBO) is used. Each particle is represented as a single element stored in an SSBO. This is a memory reserved on the graphics card. Each member has a position and a velocity that are updated by a compute shader that reads the current values from one buffer and writes the result into another buffer. That buffer is then bound as a vertex buffer and used as an instanced input to the rendering vertex shader. The algorithm then iterates, starting again with the compute shader, reusing the positions and velocities calculated in the previous pass. No data leaves the GPU memory, and the CPU is not involved in any calculations. An alternative to pass data to the GPU is via buffer textures that are used with image load and store operations. A particle system can also be implemented using a transfer feedback buffer which is implemented using vertex and geometry shader programs.

Photorealistic

An alternate definition of photorealism is given by Ferwerda [27]. The author considers photorealism as images that are photometrically realistic, where *photometry* is the measure of eye's response to light energy. Thus, photorealistic rendering is about simulating light or how photons move around in a scene. The better the approximation of this process, the closer we can get to photorealism. Techniques like ray- or path-tracing, which simulate photons bouncing around in a scene, are inherently better at producing photorealistic results. However, these techniques do not to work very well for a dynamic scene with many thousands of moving particles. The implementation is further compounded by stereoscopic output at video frame rates, where a minimum of 120 fps screen refresh is required to achieve a jitter free stereo animation of rain or snow. Thus, an approximation to photorealistic results is desired. Fortunately, it is difficult for the human eye to notice subtle differences in a dynamic scene. Therefore, ignoring certain photorealistic effects, such as soft shadows, which may make a visual difference in an otherwise static scene, is a viable option.

The fragment shader programs are used to create realistic illumination effects using a GPU. The image-based light and environment mapping techniques are used to reflect and refract light from scene objects including rain and snow particles. Cube maps are one commonly used variant of environment mapping which maps the reflection and refraction vectors from the surrounding texture on to the particle. The benefit of this approach is that it requires less computation and is therefore good for realtime applications. However, the problem with this approach is that it only deals with the front surface, not the back or any other intersecting polygons in the main object. This means anything between the front face of the particle and the background is not considered. Thus the reflection and refraction of any moving objects will not appear in the particle; only the static surrounding will be reflected and refracted.

For simple monoscopic scenes, the cube map approach can give some interesting results. However, to get more realistic stereo results it requires multiple passes that create a new cube map during each frame cycle. The other background objects in the scene are rendered to an off-screen buffer, which is used as a cube map for scene illumination. The stereo implementation of cube maps is performed for both the left- and the right-eye views. This potentially doubles computation.

OpenGL is a rasterization-based system but there are other methods for generating images such as ray-tracing. To take realism and global illumination into account, a move towards ray-tracing is necessary. The ray-tracing simulates natural reflection, refraction, and shadowing of light by 3D surfaces. In general ray-tracing is computationally expensive due to many calculations required to determine object-ray intersections. Parallel implementation using a GPU are ideal for ray-tracing based image generation. In ray-tracing, since pixel color is determined by tracing a ray from the viewpoint towards the light source, this approach inherently avoids hidden view in a stereoscopic image and can be computed with as little as five percent of the effort required to fully ray-trace surface issues [28]. Distributed parallel ray-tracing implementation is best suited using a compute mode programming interface where GPUs are utilized as general purpose processors for speed up. The realtime output from ray-tracing for an animated scene with thousands of moving particles such as in rain or snow natural phenomenon is an open area of research.

Stereoscopic

The left- and right-eye views are calculated and rendered separately. Thus, the scene is rendered from two different perspectives. The scene with rain or snow particles is modeled using the world coordinates system while each object is modeled using their respective model coordinates. Several matrix transformations are required to transform all model coordinates to screen coordinates. This approach is typical in a raster graphics pipeline where scene objects are used to form an image on the screen. The implementation is similar to methodology described by Hussain and McAllister [3], where precipitation is rendered within a bounding box. The farther away rain or snow particles from the stereo camera, the smaller the disparity in the two views. A particle at infinity will have zero disparity. If the inter-axial distance, the distance between the left- and the right-view camera, is kept similar to the distance between the human eyes, which is about 6.5cm, then the stereovision will only be effective up to 30m from the camera [29]. Since stereovision will only be strong for precipitation forming close to the camera, to improve rendering speed we can adjust the precipitation bounding box such that any particle formed outside of this boundary need only be rendered once. Conversely, the inter-axial distance can be increased to produce hyper-stereo output suitable for viewing in stereo outdoor scenery and distant rain or snow particles.

The OpenGL 4.5 graphics library is used for implementation. The application executes on the Intel Quad Core i7 CPU running at 2.00 GHz with 16.0 GB of RAM installed with an NVIDIA Quadro

K5000 GPU. The graphics card used in this study supports advanced and the most modern graphics pipeline featuring a programmable GPU. The hardware used for experiments supports quad buffer stereo with active shutter glasses to view stereo output. Quad buffering is a technology for implementing stereoscopic rendering that uses double buffering with a front and back buffer for each eye, totaling four frame buffers. Quad buffering allows swapping the front and back buffers for both eyes in sync, allowing the display to seamlessly work with different rendering frame rates. This type of stereoscopic rendering requires a display refresh rate of at least 120 Hz. This leaves 0.833 ms to complete writing data to the framebuffer. A GPU that supports quad buffering also supports time-sequential images to be viewed by active glasses with a liquid crystal display (LCD). The glasses used in this method do not have filters. Instead an LCD acts as a blocking shutter. An electronic signal is used to either make the lenses clear or opaque. The signal alternates for each eye causing the left eye to see the view while the right is blocked and vice versa. The view from the glasses is actively synchronized with the current frame on the display via an infrared signal between the active shutter glasses and the display. This method is capable of delivering a full high resolution progressive image to each eye. Since active shutter glasses do not use polarized light or color filters, the light intensity reaching the viewer is higher. However, the major disadvantage of this system is that is requires additional logic to maintain synchronization between the display and the glasses.

Alternatively, the *frame buffer object* (FBO) can be used to produce stereo anaglyphs. In a fragment shader, instead of writing the pixel data to the frame buffer for rendering, the FBO is used to save the pixel data. The fragment shader generates stereo anaglyphs by rendering left- and right-eye images on separate FBO that are rendered to texture to produce a composite of left- and right-eye views. The red component comes from the left-eye image while the green and blue components come from the right-eye image. Recent techniques for computing anaglyphs that improve color faithfulness which are based on the transmission properties of the filters and the color characteristics of the display device have been proposed. The anaglyph output can be improved by using the uniform approximation algorithm to produce brighter output [30]. The CIELab approximation method can preserve color fidelity [31]. However, such algorithms incur significant extra computational overhead in anaglyph calculation.

Experiments and Results

There is no standard method to measure photorealism [32]. The images used in the experiments and the test procedure itself can cause variability in the results. Therefore, new experiments will have to be devised to measure photorealism depending on the test scenarios. In this study, answers to survey questions are collected as data. Participants are shown video clips and still photographs of natural scenery with rain and snow. The visual stimuli vary along three factors or dimension that forms the perceptual space of a precipitation scene. From the literature review of related work done in computer generation of rain and snow it is concluded that there are three key factors influencing perceptual space, thus important for producing photorealistic results; they are number, size, and movement particles. The experiments performed in this study measure the statistical ranking and importance of the three visual factors. Additionally, computer rendered natural scenes with precipitation vary in light conditions such as precipitation during sunlight vs. overcast sky, glow or halo from artificial light sources such as around streetlamps, and fog or atmospheric haze effects. Participants also evaluate stereoscopic views of computer generated natural scenery with precipitation by answering survey questions designed to compare monoscopic and stereoscopic outputs.

The perception of each person varies slightly, thus making visual fidelity extremely subjective. Therefore, human subjects are asked to complete survey questions from which conclusions can be drawn about photorealism. Approximately 5 to 10 percent of the general population is stereo blind and cannot sense depth therefore it is important to screen the participants for stereovision. The International Telecommunication Union (ITU) has proposed recommendations for performing stereovision tests for participants in subjective assessment [33]. Guidelines from the ITU recommendations help with screening subjects for visual acuity and stereo blindness. A total of sixty healthy adult subjects, who are not stereo blind, participated.

Three different types of experiments are conducted. In the first type of experiment, a set of questions are designed to determine the perceptual space of precipitation in terms of number of particles, size, and their motion as they fall towards ground. The second experiment is designed to evaluate other visual factors such as illumination, fog, and glow effects in a computer generated precipitation scene. In the final experiment, a question is asked to compare mono and stereoscopic rendered outputs to study the contribution of stereo on photorealism. The data gathered from these experiments is analyzed by using statistical tools.

Perceptual Space

The perceptual space is defined as the visual experience of a precipitation scene as observed from the ground. The perceptual space is influenced by visual factors, such as particle sizes and number of particles that determine the precipitation intensity - light, moderate, and heavy rain or snowfall conditions. Additionally, variations in particle motion such as vertical or random motion in case of snowfall that exhibits greater variation in vertical motion as compared to rain and slanted particle motion due to wind effects or turbulence is an important visual factor to consider. These factors are ranked based on the result of series of survey questions asked in controlled human subject experiments to determine relative influence of a visual factor on the perceptual space of a rain or snow event. For example, if respondent is most sensitive to variations in the number of particles then, to enhance photorealism, a rendering algorithm can be developed to emphasize this particular visual factor.

The input stimuli are a series of video samples and still pictures of actual rain and snowfall scenes captured by a monoscopic camera. These samples are gathered from various freely available public websites [35]. The stimuli vary along the three factors such that each factor varies in extreme between high and low values. This results in a total of eight input stimuli, varying in size, number of particles, and particle motion as they fall. The quantities of high and low values are subjectively judged after visually inspecting and comparing several input stimuli. For example, an image of a light rain scene will be selected as one of the input stimulus for having very few and small rain streaks falling vertically after comparing it will several similar images. Figure 1 shows a sample image used to show heavy rain with many particles falling on a slant used as an input stimulus for one of eight experiments. A similar visual stimulus for experiment with snow is also shown. Note that the included images lose resolution and visual fidelity in rain and snow particles after a resize.

Each input stimulus is used in an experiment that asks thirty subjects six questions about the three visual factors. The Likert scale

is a commonly used scale to rank human responses to survey questions. Responses to the six questions, which are referred to as Likert items, are recorded on a five-point Likert scale [36].

Figure 1. One of the eight visual stimulus for rain and snow scene [37]

The participants are shown input stimuli until they complete their responses to all six questions. There are two questions per visual factor forming a total of six questions as listed in Table 1. Note that input stimuli with particle attributes of "small" vs. "large", "few" vs. "many", and "straight" vs. "angle" are subjectively selected for these experiments after visually inspecting several samples.. All even numbered questions are opposite to the previous odd numbered questions. This identifies incorrect survey answers due to *respondent bias*, which is participants' inability to answer truthfully or accurately. The participant response time is neither restricted nor measured. It is also important to note that there is no notion of "correctness" as the responses are subjective. Since Likert scale responses produce ordinal data with no clear distribution, the statistical analysis such as means and standard deviations are not useful for analysis. For example, it is unclear what the average of "strongly agree" and "strongly disagree" means. Instead calculating median or measuring frequency of responses in each category provides a more meaningful analysis [38]. A different population set of participants is used to repeat the above experiments for visual stimuli that have snow precipitation instead of rain.

Table 1: Survey questions to determine perceptual space

Instructions: Answer each question by writing a number between 1 and 5. Note: 1 = strongly disagree, 2 = agree, 3 = Neutral, 4 = agree, and 5 = strongly agree	
No.	Questions
1	Particle size is small
2	Particle size is large
3	There are few particles
4	There are many particles
5	Particles are falling straight down
6	Particles are falling at an angle

A total of thirty respondents participant in eight experiments and answer six questions per experiment for rain scenes. For each experiment a participant answered two questions per visual stimulus such that one question is oppositely worded from the other to detect respondent bias. The results from all experiments are collectively analyzed except for questions that were designed to detect respondent bias. Figure 2 show such plots to compare the descriptive statistics, such as median, range and inter-quartile range (IQR), of the three visual factors.

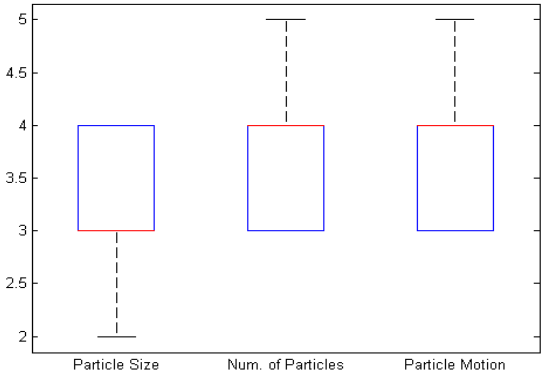

Figure 2. Rain - ranking of visual factors

A box and whisker plot graphically represents several numeric quantities [39]. The vertical axis represents the Likert scale ordinal values, from "strongly disagree" represented by rank value of 1 up to "strongly agree" with rank value of 5. The box itself represents the first and the third quartile of the data, which is the inter-quartile data range, where the majority of the response exists. The red line represents the median response. The whiskers are represented by the dashed lines showing the possible range of user response. For example, for particle size, responses ranged from "disagree" to "agree" with median response of "neutral, while the majority of responses where between "neutral" and "agree".

The IQR values show how precise data is by eliminating the outliers. For number of particles and motion, the respondents tend to favor "agree" or "strongly agree" with average response rank between "neutral" and "agree". The central tendency of both these visual factors is indicated by median response of "agree". Although the average response of particle size is also between "neutral" and "agree", the median is "neutral" with response ranging from "disagree" to "agree". The data shows that respondents have a favorable opinion regarding number of particles and motion in a rain precipitation scene but the same cannot be said about the particle size. Moreover, as shown in Table 2, the mode value for particle size is a neutral opinion as the most frequently occurring response. On the other hand, the mode of the other two visual factors are more towards "agree" or "strongly agree".

The experiments are repeated with another group of thirty respondents for snow scenes. The experiments on snowfall as visual stimuli yields similar results. Table 2 shows the median and mode values in comparison with rain as visual stimuli. In case of the snow, it is noted that the mode is "agree" for particle size as opposed to "neutral" for rain. The results suggest that the respondents note a change in particle size for snowflakes more frequently. This may be attributed to relatively lower velocity of falling snow particles, as compared to raindrops, giving viewer opportunity to observe snowfall in more detail.

Table 2: Descriptive statistics – rain/snow (perceptual space)

	Median		Mode	
	Rain	Snow	Rain	Snow
Size	3	3	3	4
Particle Numbers	4	4	4	4
Particle Motion	4	5	5	5

Other Visual Factors

In these experiments survey questions are designed to measure subjective preferences of different lighting and atmospheric conditions. During natural rain or snowfall, the sky is overcast and light is diffused. The question is whether simulating overcast lighting contributes towards improvement in photorealism of a rendered precipitation scene. The participants are asked to rate two different light conditions, daytime and simulated overcast light conditions. Other visual factors, such as glow from an artificial light source and atmospheric haze or fog effects, are also rated for photorealism.

A total of three visual stimuli and three survey questions are evaluated by thirty participants. The questions are listed in Table 3. Keeping the definition of photorealism in mind, participants are asked to view computer generated rain animations in 2D. They are first shown a rain scene in bright daylight conditions. The output is switched to show the same scene with lighting appropriate for an overcast sky. The participants are then asked to compare the two outputs. A Similar procedure is used to compare a rain scene with glow or halo from an artificial light source and a scene with atmospheric haze or fog.

Table 3: Survey questions for other visual factors

Instructions: Answer each question by writing a number between 1 and 5. Note: 1 = strongly disagree, 2 = agree, 3 = Neutral, 4 = agree, and 5 = strongly agree	
No.	Questions
1	Daylight vs. Overcast: Overcast light made scene appear photorealistic
2	Glow: Adding a glow effect made scene appear photorealistic
3	Fog: A fog effect improved photorealism

The experiments are repeated for snow precipitation with another group of thirty participants. The impact of these visual factors on photorealism is investigated by analysis. The survey questions used for these experiments compare the two visual stimuli shown in sequence in the three experiments. The comparison is made between two different light conditions, daylight vs. overcast, presence or absence of glow from simulated manmade light source, and addition of fog effect in the scene. The participants use the Likert scale from 1 to 5, "strongly disagree" to "strongly agree", to rank their responses. The results of the experiments are summarized in Figure 3.

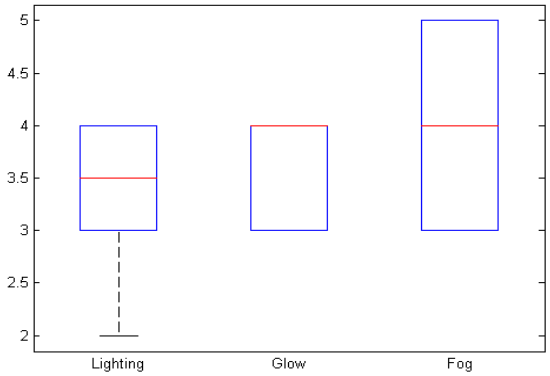

Figure 3. Rain - ranking of effects, such as light, glow and fog

Lighting plays an important role in producing photorealistic results. These experiments indicate that for a rain scene the overcast lighting produces close to neutral response as compared to the other two visual factors. The participants' response to the fog effect was most significant. The response to snow as visual stimuli is "neutral" for light conditions and glow with favorable response to fog effect. Since snow particles are opaque and lack light refraction they make a scene appear bright. The respondents did not form any strong opinion about a snow scene except to "agree" on fog contributing towards photorealism. Table 4 compares the median and mode values between the two types of visual stimuli.

Table 4: Descriptive statistics - rain/snow (other factors)

	Median		Mode	
	Rain	Snow	Rain	Snow
Lighting	3.5	3	4	3
Glow	4	3	4	3
Fog	4	4	4	4

Photorealism in Stereo

A computer generated rain scene is used in this experiment that is made up of a noticeable number of raindrops falling at slightly slanted angles with random variation in sideways motion. The scene is rendered in both mono and stereo. The subject views 2D output before viewing the same scene in stereo. The stereo is viewed by using active shutter glasses while the display is switched to produce stereoscopic output. The participants are asked to compare the two outputs. The survey questions are designed to find whether stereoscopic results appear more photorealistic relative to monoscopic output. The experiment is repeated for snow

precipitation. A same question is asked to two independent groups of thirty participants with one group answering the question with rain as the visual stimulus while the other group is asked to respond to same question with snow as the visual stimulus. The question compares a monoscopic stimulus with a stereo equivalent. The participants use the Likert scale from 1 to 5, "strongly disagree" to "strongly agree", to rank their responses.

For both input stimuli, rain and snow, responses are closer to "strongly agree". The snow scene has median and mode values slightly higher, median 4.5 and mode 5, as compared to 4 and 5 respectively for rain. The response frequency of each Likert level, for which there was a response, is shown in Table 5. Note that the majority of responses are either "agree" or "strongly agree" for both types of precipitations, which is more pronounced for snow.

Table 5: Response frequency for Likert levels

Question: Viewing in stereo made the scene appear photorealistic.	Strongly Agree	Agree	Neutral
Rain	37%	33%	30%
Snow	50%	40%	10%

With respect to the question regarding photorealism of mono vs. stereo output for rain and snow visual stimuli, the *Mann-Whitney U-test* is used to compare the responses of the two independent groups. This test is well suited to analyze the Likert scale data, which is ordinal or ranked scale, as we cannot presume that the responses fit a parameterized distribution. This test requires that results from one experiment do not affect results in the other and since the responses for rain and snow experiments are from different population groups, the two group results are independent.

Since rain and snow precipitation are similar, we want to test whether there is a statistically significant difference in respondents' opinions regarding the question posed in this experiment. We can perform a hypothesis test by defining the null and an alternative hypothesis. The null hypothesis is that there is no difference, using a significance level of 0.05, between the outcome of the two experiments using either rain or snow as visual input when it comes to comparing mono and stereoscopic output. In other words, both groups are expected to respond similarly regardless of the type of the visual stimuli, rain or snow, 95% of the time. The alternative hypothesis is opposite of the null hypothesis, there is a difference between the outputs of the two experiments.

An online Mann-Whitney U-test calculator is used to test the null hypothesis [34]. Raw samples are entered as an input to the calculator. The results confirm our null hypothesis by calculating the *Z-score* of -1.45627 and the *p-value* of 0.1443. Since p-value is greater than our significance level of 0.05, we will accept the null hypothesis, concluding that both groups formed similar opinions that viewing in stereo makes the scene appear photorealistic regardless of precipitation type.

Conclusions

We conclude that in terms of the particles size, number of particles, and motion in a precipitation scene, visual stimuli of either rain or snow generate the same response. Both types of precipitation generated stronger opinions for the number of particles and motion suggesting that these two visual factors have more effect on participant's attention. Thus the rendering algorithm should emphasize these visual factors to enhance visual realism. The results demonstrate that the visual factors for photorealism can be ranked as more sensitive to number of particles and motion than to size.

In studying other visual factors, such as lighting conditions, glow, and fog effect the results from the two visual stimuli differ slightly. As expected rain precipitation was more sensitive to these factors while responses to snow scenes had fewer variations in opinion, remaining close to neutral. The overcast sky condition for rain scenes produced responses slightly higher than neutral suggesting that participants still considered rain as visually real even in normal daytime light conditions. However, presence of atmospheric haze and fog produced a stronger response; rendering outdoor scenes with fog effects adds to realism.

Since this part of experiment used a 2D visual stimuli, the glow and fog effects contribute towards photorealism independent of stereo. Moreover, the stereoscopic output contributed towards photorealism when compared to monoscopic results. The median response for a snowfall scene is slight higher than rain scene. This can be explained with snow particles falling at much slower rate than rain resulting in more time to observe particles in stereo. The survey can be enhanced to consider other questions, such as stereo output and visual immersion for effectiveness of stereoscopic view in virtual reality applications, the impact of glow effects on photorealism to scene with nighttime ambient light, and particle interaction with other scene objects, such as snow accumulation or raindrop splash effect on photorealism. In these experiments we asked a single question about apparent photorealism in stereo. A future enhancement to this experiment is to improve comparison of a mono to a stereoscopic visual stimulus, such that we gather opinions from monoscopic visual stimulus and evaluate it against opinions from same output in stereo. Stereoscopic implications of camera attributes, such as lens distortions on photorealism is an open area of research. Future research will use methods to exploit the geometry of stereo pair to speedup rendering of photorealistic scenes with natural phenomena while maintaining realtime frame rates.

References

[1] J. Peixoto and A. Oort, Physics of climate, New York, NY: American Institute of Physics, 1992.

[2] Q. Zhao, "Data acquisition and simulation of natural phenomena," *Science China Information Sciences,* vol. 54, no. 4, p. 683–716, 2011.

[3] S. Hussain and D. McAllister, "Stereo rendering of rain in real-time," in *Proc. SPIE*, San Francisco, CA, 2013.

[4] P. Rademacher, J. Lengyel, E. Cutrell and T. Whitted, "Measuring the Perception of Visual Realism in Images," in *Proc. Eurographics*, London, UK, 2001.

[5] B. Rose and D. McAllister, "Real-time photorealistic stereoscopic rendering of fire," in *Proc. SPIE*, San Jose, CA, 2007.

[6] J. Borse and D. McAllister, "Real-time image-based rendering for stereo views of vegetation," in *Proc. SPIE*, San Jose, CA, 2002.

[7] T. Johnson and D. McAllister, "Real-time stereo imaging of gaseous phenomena," in *Proc. SPIE*, San Jose, CA, 2005.

[8] C. Zhang and T. Chen, "A survey on image-based rendering - representation, sampling and compression," *Signal Processing: Image Communication,* vol. 19, no. 1, pp. 1-28, 2004.

[9] S. Starik and M. Werman, "Simulation of Rain in Videos," in *Texture Workshop, ICCV*, Nice, France, 2003.

[10] Y. Mizukami, K. Sasaki and K. Tadamura, "Realistic Rain Rendering," in *GRAPP*, Madeira, Portugal, 2008.

[11] K. Garg and S. Nayar, "Photorealistic Rendering of Rain Streaks," *ACM Trans. on Graphics,* vol. 23, no. 3, pp. 996-1002, July, 2006.

[12] L. Wang, Z. Lin, T Fang, Y. Xu and S. Kang, "Real-Time Rendering of Realistic Rain," *ACM SIGGRAPH Sketches,* p. 156, 2006.

[13] C. Creus and G. Patow, "R4: Realistic Rain Rendering in Realtime," *Computers & Graphics,* vol. 37, no. 2, pp. 33--40, 2013.

[14] C. Wang, M. Yang, X. Liu and G. Yang, "Realistic Simulation for Rainy Scene," *Journal of Software,* vol. 10, no. 1, pp. 106-115, 2015.

[15] N. Wang and B. Wade, "Rendering Falling Rain and Snow," *ACM SIGGRAPH 2004 Sketches,* p. 14, 2004.

[16] Y. Yang, X. Zhu, J. Mei and D. Chen, "Design and Realtime Simulation of Rain and Snow based on LOD and Fuzzy Motion," in *ICPCA*, Alexandria, Egypt, 2008.

[17] P. Fearing, "Computer modelling of fallen snow.Proceedings," in *Proc. of the 27th annual conference on Computer graphics and interactive techniques*, New Orleans, LA, 2000.

[18] B. Feldman and J. O'Brien, "Modeling the Accumulation of Wind-driven Snow," in *SIGGRAPH 2002 Conference Abstracts and Applications*, San Antonio, TX, 2002.

[19] R. Fedkiw, J. Stam and H. Jensen, "Visual Simulation of Smoke," in *Proc. of the 28th annual conference on Computer graphics and interactive techniques*, Los Angeles, CA, 2001.

[20] C. Zou, X. Xie and G. Zhao, "Algorithm for generating snow based on GPU," in *Proc. ICIMCS*, Harbin, China, 2010.

[21] N. Festenberg and S. Gumhold, "A Geometric Algorithm for Snow Distribution in Virtual Scenes," in *Eurographics Workshop on Natural Phenomena*, Munich, Germany, 2009.

[22] N. Tatarchuk and J. Isidoro, "Artist-Directable Real-Time Rain Rendering in City Environments," in *Proc. NPH*, Vienna, Austria, 2006.

[23] H. Pruppacher and J. Klett, Microphysics of Clouds and Precipitation, Dordretch, Netherlands: Springer, 1996.

[24] C. Magono and W. Lee, "Meteorological Classification of Natural Snow Crystals," *Journal of the Faculty of Science,* vol. 4, no. 3, pp. 321-335, 1966.

[25] R. Srivastava, "Size distribution of the raindrops generated by their breakup and coalescence," *Journal of Atmospheric Sciences,* vol. 28, no. 3, pp. 410-415, 1971.

[26] W. Reeves, "Particle Systems -- a Technique for Modeling a Class of Fuzzy Objects," *ACM Trans. Graph,* vol. 2, no. 2, pp. 91-108, 1983.

[27] J. Ferwerda, "Three varieties of realism in computer graphics," in *Proc. SPIE*, Santa Clara, CA, 2003.

[28] S. Adelson and L. Hodges, "Stereoscopic ray-tracing," *The Visual Computer,* vol. 10, no. 3, pp. 127-144, 1993.

[29] C. Ware, "Dynamic Stereo Displays," in *Proc. SIGCHI*, Denver, CO, 1995.

[30] Z. Zhang and D. McAllister, "A uniform metric for anaglyph calculation," in *Proc. SPIE*, San Jose, CA, 2006.

[31] D. McAllister, Y. Zhou, and S. Sullivan, "Methods for computing color anaglyphs," in *Proc. SPIE*, San Jose, CA, 2010.

[32] S. Højlind, C. Schnatterbeck, J. Klein, L. Østergaard, R. Arberg, S. Nadarajah, M. Hansen, and M. Kraus, "Why a single measure of photorealism is unrealistic," in *Proc. SCCG*, Smolenice, Slovakia, 2014.

[33] ITU, "BT.1438 : Subjective assessment of stereoscopic television pictures," ITU, 25 6 2015. [Online]. Available: https://www.itu.int/rec/R-REC-BT.1438-0-200003-W/en. [Accessed 15 12 2016].

[34] J. Stangroom, "Social Science Statistics," 2017. [Online]. Available: http://www.socscistatistics.com/tests/Default.aspx. [Accessed 11 1 2017].

[35] "Wikimedia Commons," MediaWiki, 10 1 2017. [Online]. Available: https://commons.wikimedia.org. [Accessed 11 1 2017].

[36] R. Likert, "A technique for the measurement of attitudes," *Archives of Psychology,* vol. 22, no. 140, pp. 1-55, 1932.

[37] L. Micheler, "Snowfall in Germany," Wikimedia, 26 2 2016. [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=47212597. [Accessed 11 1 2017].

[38] S. Jamieson, "Likert scales: how to (ab) use them," *Medical education,* vol. 38, no. 12, pp. 1217-1218, 2004.

[39] N. Cox, "Speaking Stata: Creating and varying box plots," *The Stata Journal,* vol. 9, no. 3, p. 478–496, 2009.

Author Biography

Syed Hussain completed his undergraduate and graduate studies in computer and electrical engineering in 1998 from NC State University. After graduation he worked as a software engineer for ten year. Currently he teaches at Wake Technical Community College while working towards a PhD in computer engineering at NC State.

David F. McAllister obtained his BS in mathematics from University of North Carolina Chapel Hill in 1963. He then spent two years onboard the USS Enterprise, at the end of which he entered Purdue University where he obtained his MS in mathematics in 1967. He taught mathematics at UNC Greensboro while working for his PhD in computer science at UNC-CH. He completed his PhD in 1972 and joined the computer science faculty at NC State.