Abstract
In this paper we propose the utilization of the X3D standard resources and flexibility to visually improve the observation of electromagnetic fields propagation by providing the simulation results through an X3D animated scene. Although the examples apply to the specific topic of electromagnetic waves, the concepts presented in this work could be adapted or extended to other scenarios and applications of simulation. The x3d animated scene presented in this paper allows the interested parties to easily evaluate the simulation snapshots and have access to important interaction techniques like rotation, translation, zoom and time setting, going back and forth over the simulation steps.

1. Introduction
Graphical representation of data, in most cases, is one of the best ways to understand the information gathered from some scientific process. It’s commonly used to analyze the results of a simulation or simply observe the representation of physical phenomena. Even more, visualizations have been found to be a valuable complementary pedagogical tool in teaching all sorts of topics such as biology, oceanography, chemistry, physics etc.

In practice, sharing the results of a simulation or the representation of physical phenomena reveals to be very complex, time consuming, boring and often incomplete. Most of the time, these results are shared to the interested parties by using tables, two-dimensional graphics, pictures or movies. However, all these alternatives might be restrictive somehow. For instance, when evaluating the results of a simulation, a movie might probably hide some important details that would be available in a different point of view, thus avoiding a complete analysis of the generated result.

2. Previous work
Previous work have already been developed to demonstrate the application and efficiency of x3d as a powerful complementary pedagogical tool. The work recently presented by Chittaro and Ramon [4] provides a deep analysis on the advantages and disadvantages of using Virtual Reality and Web3D technologies in education. One the most important ideas presented by the authors is the ability provided by the x3d standard to easily share interactive and educational content, in contrast with proprietary software and hardware involved when deploying common Virtual Reality solutions.

These advantages have already been explored by some educational projects with the aim of teaching all sorts of topics like chemistry [5], medicine [2] and cultural heritage [3].

The teaching of abstract topics, like electromagnetic field propagation, can be tremendously facilitated by a graphical representation of the physical phenomena. For instance, by supplying a group of students with a software for visualizing and animating electromagnetic fields, significantly higher understanding was achieved as opposed to a
reference group that was taught in a traditional lecture format [6].

In this paper, the simulation of electromagnetic field propagation is solved to a few predetermined time steps. In this case, it is very important to provide an X3D animated scene which allows the students to repeat the animation as many times as desired, going forth and back over the simulated time steps, along with other basic interaction features like rotation, translation and zoom.

With the aim of providing an X3D scene with all the mentioned features, it is important to consider some previously developed work at improving the control of animations in an X3D scene. The nodes proposed by Belloc et al. [1] presents a clever way of allowing time control over animations. Besides that, there are also some resources used to display interactive menus, allowing the user to easily manipulate the evaluation of animations.

3. Simulation

The simulation focuses on electromagnetic waves used in networks of the type IEEE 802.11 (Wifi) and the visualization of the interaction between a wave and typical objects that it might interfere with. The mathematical model used is derived directly from Maxwell’s equations and thus, automatically models physical effects such as reflection, attenuation and diffraction. Also, this model works in the time domain and can therefore be programmed to return the time-varying volumetric data continuously or at certain time intervals.

3.1. Finite-Difference Time-Domain method

The standard Finite-Difference Time-Domain (FDTD) method was first presented by Yee in 1966 [9]. The FDTD method uses centered finite differences to approximate Maxwell’s equations in both the time and space domain. The basic idea of FDTD is to first update the time derivative of the electric fields by use of the magnetic field values, and then update the time derivative of the magnetic field values using these new electric field values. This process is then repeated until the electromagnetic wave has been advanced a certain distance from the antenna, or until steady state is reached.

3.2. Extraction of Volumetric Data

To complete one simulation step in the time domain, the FDTD algorithm basically runs through all the points in the discretized volume and first updates all the electric field values and then all the magnetic field values. However, there are actually six volumes of scalar data that are being calculated at all simulation steps, one for each of the three electric field values, $E_x, E_y, E_z$ and similarly for the magnetic field values, $H_x, H_y, H_z$.

To be able to visualize a propagating electromagnetic field, these six volumes have to be merged into one single volume. This can be done by first calculating the Poynting vector, $P = E \times H$ [W m$^{-2}$] of the electromagnetic field for all points in the data grid, and then take the absolute value of these vectors to obtain scalar values that represent the power density at those points. These calculated scalar values will now serve as the basis for the extraction of the iso-surfaces by applying the Marching Cubes algorithm.

3.3. The Marching Cubes algorithm

A propagating electromagnetic pulse can be represented by an advancing wave front. To create such wave fronts from the volumetric data set, the Marching Cubes algorithm [7] was used to produce isovalued polygonal surfaces from the three-dimensional data set containing the absolute values of the Poynting vectors.

The Marching Cubes algorithm has demonstrated to be fast and efficient, becoming one of the most common approaches to computing iso-surfaces from discrete volumetric data. It’s application in this work have been proved to be tremendously important to simplify the simulation result. The huge three-dimensional data set generated for each time step of the electromagnetic simulation is reduced to a predetermined number of polygonal iso-surfaces. The resulting data size and quality offers the possibility of creating a much more pleasant way of delivering this kind of simulation to students.

4. Visualization

The visualization of the previously generated isosurfaces as independent X3D files is not the most appropriate way to analyze the time-varying results. Despite the fact that these separated scenes already enables simple interaction techniques such as rotation and translation, the student is restricted to the observation of a single snapshot of the result.
turning the evaluation of all the generated files into a very unpleasant task.

4.1. TimeClock and TimeMenu Nodes

The TimeClock node was proposed to overcome a few limitations of the X3D standard [1]. The standard states that the TimeSensor node, the current alternative for feeding Interpolators, is currently constrained to the evaluation of the system clock. Thus, it is not possible to go forth and back, or choose which interval of the interpolation to repeat.

These features are highly desired when analyzing time-varying events. In order to make these features easily accessible to the students, a simple DVD-like interface was also used along with the TimeClock node. This interface is known as the TimeMenu node, and it can be used through the extern prototype declaration. The TimeMenu node encapsulates a TimeClock node instance and allows the user to have full control over a specified animation. The TimeMenu can be easily configured as a HUD (Head Up Display, see figure 1), to be constantly available to the user.

This new created scene, also known as the evaluation scene, includes all other X3D files by using the Inline node [8]. The Inline node embeds in the current scene the file mentioned in the url field. The Switch node is then used to select which of the included files will be displayed to the user. The Switch grouping node traverses only one of the nodes specified in the children field. The traversed child is determined by the index number in the whichChoice field of the Switch node.

Therefore, to create an animation that illustrates the simulation result, it’s necessary to constantly change the traversed child of the Switch node, thus, walking through all the children and displaying the X3D files associated with each time step. To deploy this animation in the final scene, the usage of two important nodes was made necessary: The IntegerSequencer node and the TimeSensor node.

The IntegerSequencer node plays an important role into converting the system time, retrieved from the TimeSensor node, to an index number for the whichChoice field of the Switch node. The behavior of the IntegerSequencer is as follow. Basically, it evaluates the input fraction field, verifying its relative position in the values of the key field and outputs a value in the keyValue field with the same relative position.

To a better understanding of the IntegerSequencer node, consider n keys $t_0, ..., t_n$, and also n keyValue values $v_0, v_1, ..., v_n$. The IntegerSequencer will evaluate a function $f(t)$ (Equation 1), where $t$ is the fraction field, fed by the TimeSensor node.

$$f(t) = \begin{cases} 
  v_0, & \text{if } t < t_0 \\
  v_i, & \text{if } t_i \leq t < t_{i+1}, \quad i = 0, 1, ..., n-1 \\
  v_n, & \text{if } t \geq t_n 
\end{cases} \quad (1)$$

In our final scene, the evaluation of the function $f(t)$ would result in the index number to be applied to the Switch node. The following simple example illustrates the application of the IntegerSequencer together with a TimeSensor and a Switch node.

```xml
<Switch DEF="switcher" whichChoice="0" >
  <Inline url="step0.x3d" />
  <Inline url="step1.x3d" />
</Switch>
</TimeSensor>
```
In this example, the file `step0.x3d` will be displayed for the first 5 seconds of the animation, and then, the Switch node will display the file `step1.x3d` for more 5 seconds. At the end, the animation will repeat due to the fact of loop enabling at the TimeSensor instance. This behavior can be concluded by simple inspection. The TimeSensor cycle time is set to 10 seconds and the IntegerSequencer is configured to set the index 0 for the first half of the cycle, and the index 1 for next half. These indexes are directly routed to the Switch node, which displays the selected file.

Despite of being a simple example, our final scene uses the same mechanism with as much time steps as necessary to include all x3d files and represent all the simulation result in a single animated scene.

To finally achieve the desired interaction features mentioned before, such as controlling the evaluation of the animation over time, going forth and back, and select the desired time step to observe, it’s necessary to replace the TimeSensor node by an instance of the TimeMenu node. The TimeMenu node will then allow the user to have more control over the animation through a simple and intuitive user interface.

5. Results

Our final x3d scene presents an interactive way for evaluating the electromagnetic fields simulation result. Using the simple interface provided by the TimeMenu node, the user is allowed to explore improved interaction techniques to control the evaluation of the animation (Figure 2).

The x3d browser used to test the developed scene was the bs Contact x3d Player version 7.1. When one of the external dependencies of the final x3d scene is not loaded correctly, the browser usually fails with a meaningless error message. Thus, it’s important to ensure that every x3d prototype dependency can be accessed at load time. The final x3d scene has only one direct external dependency, the TimeMenu node. However, the TimeMenu has its own dependencies and access to all of them should be guaranteed before loading the scene.

Another common issue is that some x3d browsers might implement a delayed load behavior of the Inline node, and this could lead to non-smooth transitions between steps. The loading of the Inline node should be determined by it’s load field (true by default). However, some browsers might ignore this field and load the scene only when the Inline node is traversed. To avoid this behavior, the x3d browser must be properly configured.

![Figure 2](image)

**Figure 2.** Evaluating the simulation steps for 1.7 ns, 2.6 ns, 4.6 ns and 6 ns

6. Conclusion and Future Work

The work developed in this paper revealed to be a very interesting and pleasant approach to disclosing electromagnetic simulation results for teaching purposes. Although the practical examples apply to electromagnetic wave propagation, the concepts presented in this work could probably be used to provide results of other time-varying physical phenomena.

A future work will consider the development of a web interface to accommodate various simulation results. It is also desired to replace the current xml encoding by the compressed binary alternative. This would reduce the amount of data transferred over the network, facilitating the disclosure of simulation results through the internet.

7. References


