Wave Propagation Using the Photon Path Map

Arne Schmitz
Computer Graphics Group
Ahornstr. 55
Aachen, Germany
aschmitz@cs.rwth-aachen.de

Leif Kobbelt
Computer Graphics Group
Ahornstr. 55
Aachen, Germany
kobbelt@cs.rwth-aachen.de

ABSTRACT

In wireless network planning, much effort is spent on the improvement of the network and transport layer – especially for Mobile Ad Hoc Networks. Although in principle real-world measurements are necessary for this, their setup is often too complex and costly. Hence good and reliable simulation tools are needed.

In this work we present a new physical layer simulation algorithm based on the extension and adaptation of recent techniques for global illumination simulation. By combining and improving these highly efficient algorithms from the field of Computer Graphics, it is possible to build a fast and flexible utility to be used for wireless network simulation. Instead of the wave nature of EM radiation we rather use the particle nature of waves to compute a discrete sampling of the volumetric electromagnetic field by tracing stochastically generated photon paths through the scene. This so called Photon Path Map is then used to estimate the field density at any point in space and also provides local information about the delay spread. The algorithm can be applied to three dimensional indoor as well as outdoor scenarios without any changes and it scales only logarithmically with the growing complexity of the underlying scene geometry.

1. INTRODUCTION

In this work we introduce an algorithm to efficiently and accurately simulate the physical layer of microwave wireless networks which is applicable both to GSM mobile phone networks and 802.11 style wireless local area networks. Especially in the case of Mobile Ad Hoc Networks (MANET) and Wireless Mesh Networks a robust simulation in the development phase is important. This is partly due to the fact that it is hard to acquire enough people and mobile stations for test runs of the algorithms. These simulations can of course also be used for antenna placement with maximum availability of deployed stationary nodes.

A typical network simulator emulates all layers, including the link or physical layer. Especially this layer contributes a lot to the computation complexity, since it is used in each query on the network medium. Thus in this work we will present an efficient method to simulate the physical layer of wireless networks.

Our algorithm efficiently computes accurate simulations of radio waves propagated through three dimensional indoor or outdoor scenes. It is derived from algorithms known from Computer Graphics, where the Global Illumination in 3D scenes is estimated for visualization purposes. This is a similar problem setting as the radio wave propagation problem, although in a different frequency band of the electromagnetic spectrum.

By approaching the problem of wave propagation from the viewpoint of Computer Graphics it is possible to develop a very general, yet accurate and fast algorithm. The only assumption regarding the scene geometry is that it is defined by a set of polygons. So any indoor and outdoor scenario is possible. As a consequence complex scenes using buildings with multiple stories are as easy to process, as scenes containing only one floor or simple geometry. Also we do not put any restrictions on the transmitter or the possible frequency range.

It is possible to incorporate our approach efficiently into a network packet simulator. This is done by computing a set of simulations in advance and doing interpolated lookups during the simulation of the network traffic.

In the following we will present related work done in this area. Then we will explain our new approach, some experimental results and finally our conclusion.

2. RELATED WORK

There has been done quite some previous work in both the areas of Radio Wave Propagation and Global Illumination. Both share similar roots and goals and this section will show on which work our new technique is based.

2.1 Wave Propagation

There are different possible approaches to wave propagation calculations. All these approaches aim at predicting the advancing of a wavefront through a scene. Upon collision with boundaries of different media the wavefront can be scattered, reflected, diffracted or refracted. Also part of the energy will

be absorbed and transformed into heat or infrared radiation. These effects will have to be taken into account to a certain degree, in order to achieve a correct solution. Here we will concentrate on algorithms that are mostly based on geometrical optics, i.e., methods that trace elementary electromagnetic waves along rays or straight line segments. Although there are algorithms known in Computer Graphics that compute wave propagation as the propagation of complex wave coefficients in a grid [16], this proved to be not efficient for small wavelengths.

2.1.1 Empirical Models

First one has to distinguish between empirical and deterministic approaches. The empirical methods use a set of parameters to model a function that as closely as possible resembles the pathloss of a signal as described by the advancing wavefront. Popular empirical methods include the Walfisch-Ikegami-Model (WI-Model) [22]. The WI-Model takes into account multiple diffraction on rooftops and is as such almost only suitable for outdoor scenarios in urban microcells. In these scenarios rooftop to street diffraction is predominant. But waveguiding effects by buildings cannot be modelled very well. So indoor scenarios and outdoor propagation paths that rely on reflection are not well modelled by this approach.

Furthermore one needs to supply well calibrated parameters to the algorithm to get sufficiently good results. On the other hand, empirical methods usually require very little computation time, since they abstract from the detailed scene geometry.

In comparison, our algorithm models reflections very well and only relies on the geometry as input.

2.1.2 Deterministic Models

Deterministic algorithms use ray tracing techniques to calculate propagation paths. In the wireless network world we distinguish between ray tracing and ray launching techniques. The same techniques are known in computer graphics, where they are also called ray tracing and light tracing, depending on the ray direction [7].

In [11] it was shown that adaptive space subdivision leads to very efficient ray-tracing results compared to uniform subdivision. By using kd-trees [4] the complexity of the 3D or 2D ray-object intersection test can be reduced to $O(\log n)$. This makes the use of raytracing algorithms efficient and also allows them to process large amounts of geometry. We use this technique to accelerate both the ray intersection and the search for the nearest diffracting edge.

In [24] a 2D algorithm is proposed with several accelerations. For a given transmitter and receiver it shoots only rays in an approximated Fresnel zone. Since this approach needs to do a raytracing step for each receiver pixel, a caching heuristic is employed to avoid recalculation of rays for neighboring pixels. The need for caching is eliminated in our approach by sampling only those areas which receive energy. There is no need anymore to evaluate the ray-tracing for each receiving voxel.

A 3D ray launching algorithm working on a uniform grid was developed by [19]. Here the rays are propagated in discrete steps on the grid and the signal is reflected, transmitted or diffracted when certain voxels are hit. The accuracy of the ray propagation thus directly depends on the grid resolution. This is not the case with our algorithm, since the Photon Path Map is computed without a discrete grid as its base.

Another approach is the dominant path prediction model which can be adapted to work in indoor [23] and outdoor [21] scenarios. Dominant propagation paths are determined in a non-trivial way by using neural networks [25], [26]. The loss along these paths is then evaluated using some empirical methods. For the indoor case waveguiding factors are taken into account, since reflection on walls is a major factor for these scenarios. The advantage of this approach is speed and relatively high accuracy. But since only dominant paths are used it is unclear if the solution is somehow biased, e.g. if it has the tendency to find local extrema. Also one has to change the algorithm to switch between indoor and outdoor scenarios. This is not necessary with our algorithm, which can even do scenes with both out- and indoor parts.

2.2 Global Illumination

Our algorithm is an extension of the Photon Map [12] which in turn can be seen as a variant of the light tracing algorithm [8]. All of them can be called Global Illumination algorithms. In the field of Global Illumination one computes the distribution of light on the object surfaces in a three dimensional scene. This distribution takes into account all possible, *global* interactions of light with objects in the scene. This way one can estimate light and shadows and render visually convincing images.

Considering that visible light and microwave radiation are essentially the same phenomenon, only at different frequencies, it is only natural to transfer some of the Global Illumination algorithms to the Radio Wave Propagation setup.

Some differences have to be noted, though. First, microwaves have a much longer wavelength than visible light and so the effect of wave diffraction is relevant, whereas for visible light this can almost always be neglected. Second, in Global Illumination most of the algorithms deal with the appearance of object *surfaces*, but for Radio Wave Propagation we need to deal with field densities in a *volume*.

2.2.1 The Photon Map

The Photon Map was introduced by [12] for global illumination purposes. It is a spatial data structure which stores incoming radiance on 2D surfaces. Radiance is a radiometric quantity that describes the amount of energy arriving at the surface of an object. The reflected radiance describes the amount of energy that is reflected from this surface. This is what we can perceive as the brightness of an object.

The advantage of the Photon Map is that the radiance can be calculated with arbitrary precision as a preprocessing step. The final energy density can be computed with some user-defined precision, allowing for efficient computation of images.

The Photon Map is computed using a two step algorithm. In a first step, the radiation is propagated in discrete quanti-

ties called photons into the scene. Each collision of a photon with objects in the scene is stored and as a result a radiance representation is produced. In the second step, this representation can be evaluated by gathering the radiance, i.e., the energy density is estimated at every surface point in the scene necessary for the final image. Both steps will be detailed in the following paragraphs.

Shooting

The goal of the Photon Map is to compute the radiance on object surfaces. Therefore we trace a number of rays from the light- or radiation-source through the scene. This is achieved by using a Monte Carlo style path tracer as widely used in the Global Illumination context. Each collision of a ray with the scene can spawn a new ray. Hence we get a sequence of rays that is called a path and which can be terminated by the Russian Roulette technique [3]. This guarantees that the path generation is unbiased, in contrast to a fixed depth cutoff. We then obtain a very simple algorithm:

```
For each light source do
      Shoot n photons in random directions
      If photon hits surface decide randomly:
                  a) Absorb photon
                  b) Store and reflect /
                  transmit photon
```

The Russian Roulette also ensures that an arbitrary number of interactions with the scene can happen, only bounded by the material properties of the objects in the scene. Most techniques used for Radio Wave Propagation allow only for a fixed, but user definable number of reflections or transmissions.

Additionally a linear attenuation term can be applied to the photon path, depending on its length to model atmospheric attenuation for different weather conditions. For this no Monte Carlo approach is used, since the volumetric attenuation would lead to a much higher variance in the final solution. But experiments have shown that this simple linear attenuation leads to good results.

Since the Photon Map uses randomly sampled paths originating from the radiation source and since the Russian Roulette favors paths along highly reflective objects, only paths that carry a lot of energy are computed. This is a very important property of the light tracing algorithm and is of particular usefulness in the context of Radio Wave propagation. In previous works, one had to eliminate paths with little or no energy and trace only those paths that are dominant in the scene [26], or one had to cache rays to avoid duplicate computation [24]. With the Photon Map we get this behavior as an inherent property of the algorithm, while keeping it simple.

Kernel Density Estimator

As a result of the photon map we have a set of points in space that mark interactions of photons with objects. The more photons there are per surface area, the more energy this surface has received, and the brighter it will appear. This way we can estimate the energy density represented by the Photon Map very easily and also in a smooth manner.

To get an idea of how bright a surface point should appear, we need an estimator of the photon density. The standard way to do this in the Photon Map method is to use some sort of kernel density estimator (KDE). In the multivariate case of dimensionality d, the KDE is defined as:

$$\hat{f} = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\|x_i - x\|}{h}\right) \tag{1}$$

The bandwidth h of the estimator determines the smoothness of the result and also the bias. However the estimator is consistent as the bias vanishes with an increasing number n of samples and a decreasing bandwidth h. The function K is a filter kernel and influences the smoothness of the solution. x denotes the current position to be rendered, and the x_i is the position of the current sample, i.e., of the current photon.

In practice it turns out that a cubic filter kernel works best for the Photon Path Map as it will produce a smooth estimation of the field density, as compared to a simple box filter for example.

Gathering

The second part of the Photon Map algorithm consists of sampling a set of precomputed photons for each surface point to be illuminated. Over these samples we can build a KDE to get an estimated value for the incoming radiance. The sample set can be efficiently gathered by taking the k-nearest neighbors for the surface point. This is only an approximation, but works reasonably well. As we will show later we can get an exact KDE for the Photon Path Map. In the classical Photon Map approach the radiance estimate still would have to be multiplied by some reflection function, but for the Photon Path Map we can take the energy density as is.

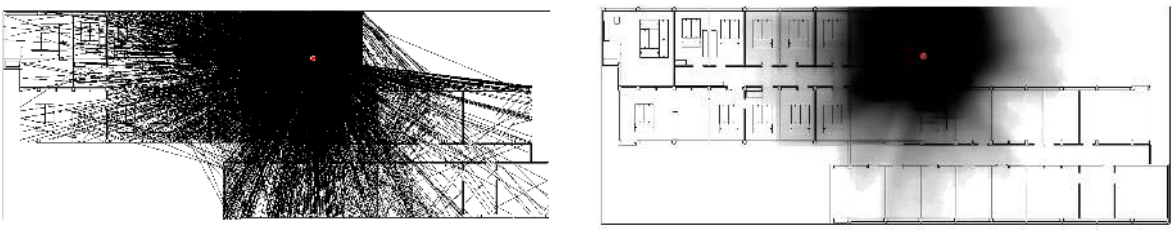
3. THE PHOTON PATH MAP

Our new algorithm extends the idea of the Photon Map to allow not only for the computation of surface radiance but also to get an estimated field strength for every point in space. Therefore we do not only store points of interaction in the map, but also the paths that connect these points as a photon travels through the scene. As a result the Photon Path Map represents a discrete sampling of paths following the advancing wavefront.

Figure 1(a) shows a Path Map for an indoor data set. Here the map is only sparsely sampled for clarity purposes and also rays that leave the scene are left out to make the image more simple.

3.1 Path Generation

The photon paths are built in much the same way as in the classic Photon Map approach. We use a standard photon mapping raytracer that is also used to generate illuminated images of three dimensional scenes. The input is given as a set of triangle meshes, surface materials per triangle and transmitters. The scene geometry is stored in a balanced kd-tree to accelerate the ray intersection tests. The output



(a) (b)

Figure 1: (a) A sparsely sampled Photon Path Map in an indoor scenario. (b) The same map, more densely sampled and with the field strength coded by using grey scales. In the black areas the field strength is high, whereas transparent or white areas have low field strength. Both images represent the top view of a 3D simulation.

of the algorithm is a volume image with the simulated field strengths per voxel.

Basically the program just runs the algorithm from Section 2.2.1 to generate photon paths, which are stored as linked lists. We will see in the next section that for this setting, since we need to take diffraction into account, we have to augment the algorithm slightly.

3.2 Ray Bending

The raytracer works as a Monte Carlo path-tracer and reflects, transmits and diffracts the rays as necessary. The diffraction of the rays is a step that is usually not needed for image synthesis. But for the simulation of radiation in the microwave spectrum this effect becomes important. Since we use a Monte Carlo approach for the sampling of the paths, we cannot use the standard Geometric or Uniform Theory of Diffraction (GTD and UTD) [13], [14]. But we can use the Heisenberg Uncertainty Ray Bending (HURB) by [10], which allows to sample the diffracted field of a ray incident on a diffracting edge embedded in a Monte Carlo path tracing process.

To find out possible diffracting edges, we store all convex edges in the scene in a kd-tree for fast lookups and decide if the current ray is bent around an edge or should be reflected or transmitted by an object. To be able to identify the diffracting edges, the input geometry should consist of manifold meshes, which can be easily guaranteed with common map data.

This leads to the following algorithm:

```
PreProcess:
    Find all convex edges and store
    them in a kd-tree

traceHURB:
    1. For a random number of steps
       2. Find nearest intersection w/object
       3. Find nearest diffraction edge
       4. Depending on order of event
          4a. Compute reflected ray, goto 1
          4b. Compute diffracted ray, goto 1
       5. Store path element
```

This algorithm can then be used in the standard Photon Map algorithm. The search for possible diffracting edges can of course be restricted to the first Fresnel zones, which depend on the simulated wavelength.

3.3 Field Strength Estimation

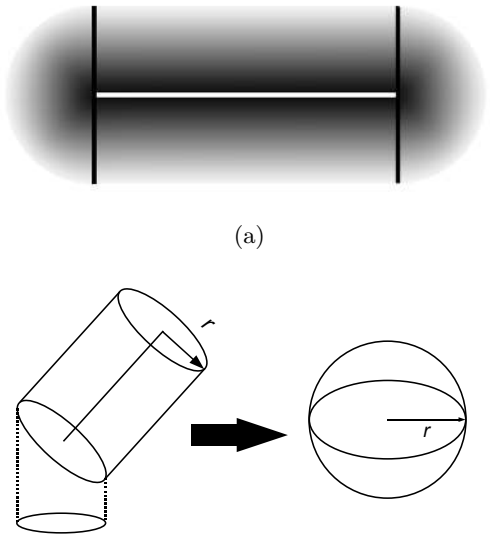
After the ray shooting we have a set of ray path segments. Now we have to determine the path density for each voxel in the output image. The kernel density estimator approach of the Photon Map requires us to find the k nearest photons to a certain point. This is relatively costly when performed on the uniform grid of the volume image. Also a k-nearest neighbor search is biased since for performance reasons the search domain has to be pruned so that only an approximate nearest neighbor search can be performed.

Instead of this we do an explicit calculation of the KDE by rasterizing the weighting function of the paths directly. The weighting function for a photon path segment produced by the kernel is similar to a distance function. For the 2D case figure 2(a) illustrates this. Because the rendering of the end caps can be slow, we can optionally reduce this to a cylindric shape as seen in Figure 2(b). This will add a small bias to the estimator but will allow us to apply a fast plotting algorithm for this cylindric weighting function. The idea is now to draw a set of parallel lines that produce a dense sampling of the cylindric shape.

The projection of the cylinder caps with an axis-aligned plane (e.g. X-Y) gives us an ellipse as shown in Figure 2(b). It is important to choose the plane so that it is as perpendicular to the cylinder axis as possible. Otherwise the intersection ellipse might degenerate.

We can then draw the cylinder by an incremental rasterization of lines that are offset according to this ellipse. The ellipse itself is defined by a center c and two axes u and v. The major axis u is always of length r, which is the cylinder radius. The minor axis is $|v| = r \cdot \sin\alpha$, where α is the angle between the cylinder caps and the major plane. The offset in the major plane is then given by the ellipse equation:

$$\langle u|(q-c)\rangle^2 + \langle v|(q-c)\rangle^2 \leq 1, \qquad (2)$$



(a)

(b)

Figure 2: (a) The falloff weighting function for a path segment (white), and the caps applied to it (black) for efficient rendering. (b) The cylinder end cap projects into an ellipse. This can be used to produce a very fast solid cylinder plotting algorithm.

where q is the current voxel. Since the cylinder base is generally inclined with respect to the major plane, we have to shift the lines perpendicular to the major plane. This displacement can then also be easily calculated by solving the plane equation for example for the Z-coordinate. This method does however produce rasterization artifacts due to the discrete nature of the line drawing algorithm. Therefore all the lines should start in an axis aligned base plane which is parallel to the major plane, but their drawing is delayed until they reach the cylinder cap. This produces a fast and solid rendering of the cylinder.

The resulting volume image represents for each voxel an approximated stationary field strength. Small scale fadings can be modelled on top of this stationary field. One solution is to use a Ricean Fading model as proposed in [17]. Also since we have computed a set of propagation paths in the Photon Path Map we can use the paths to compute an estimated delay spread, which will be detailed in the next section.

3.4 Delay spread estimation

One further advantage of our approach is that it allows the estimation of channel delay spread due to multi-path propagation. Quite a few previous works have developed models and procedures to measure and simulate the delay spread of channels [2], [20], [18]. The knowledge of these particular channel characteristics becomes ever more important for OFDM based schemes, as for example in 802.11 networks.

In contrast to the previous works, our algorithm computes a sampling of the most important paths, where we can generate a delay spread histogram for some point in space. Alternatively we compute the standard deviation of delay spread over the complete volume, which can then later be used to compute a distribution of signal delays in a simulator. Figure 3 shows an image of the standard deviation in delay spread for a small indoor scenario. Values in this image range from around 10 ns (bright areas) to 100 ns, which is in accordance with figures measured in [18] for indoor scenarios.

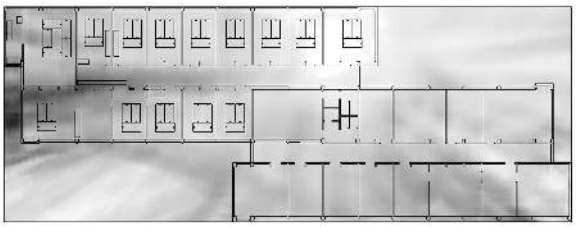


Figure 3: The standard deviation of the signal delay as a scalar value per voxel.

The results from Figure 4 confirms this observation, where we have one delay profile made from a position with line of sight to the sender, and one profile without line of sight, in a corridor. It is well known that corridors in indoor scenarios have strong multipath propagation profiles and thus a large and highly varying delay spread spectrum.

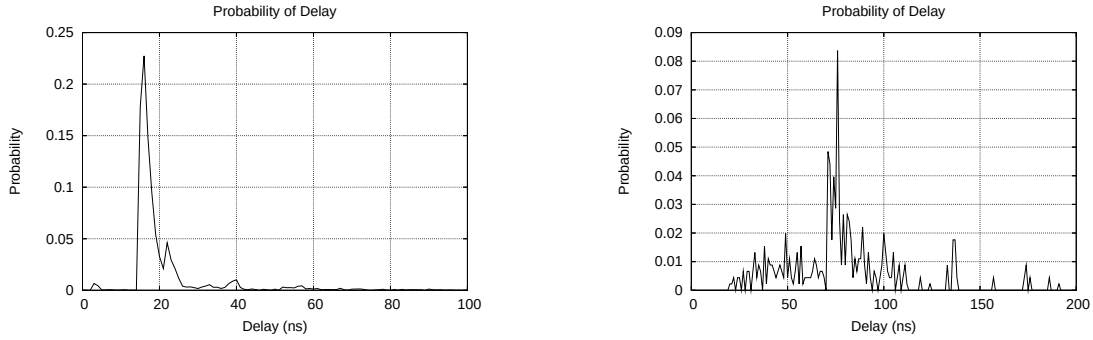
3.5 Small scale fading and interference

Small scale fading happens as a result to Doppler shift generated by moving objects in the scene. This phenomenon is not explicitly modelled by the Photon Path Map, but can be modelled by additional fading models, which depend on the underlying movement models of the simulation. This has been described in more detail by [17]. In particular small scale fadings can be induced by any moving object in the scene, which proves to be very difficult to model. For example in indoor scenarios one of the main sources for small scale fading will be the movement of people walking around.

As could be seen before, the Photon Path Map models the transmission of only one sender. In theory we can also model the transmissions of many senders, but this is not necessary in the context of the simulation of a MANET. The interference of multiple senders is being detected by the simulator (in our case ns-2), which simply does a lookup in the different Photon Path Maps for the two interfering senders. How this is all implemented efficiently will be described in the next section.

3.6 Interface with the network simulator

For the simulation of the MANET we use the ns-2 simulation package [9]. In a pre-processing step we compute for a subset of each possible transmitter position one Photon Path Map. For a typical indoor scenario of a medium sized building about 200 positions are usually sufficient. These maps can be computed on a cluster or grid network in a matter of minutes. In our experiments a grid engine of 49 Opteron computers took approximately 5 to 10 minutes to compute



(a) (b)

Figure 4: This figure shows the delay spread in the first indoor scenario. (a) Delay spread statistics for a position in line of sight to the sender. (b) Delay spread for a position without line of sight on the corridor.

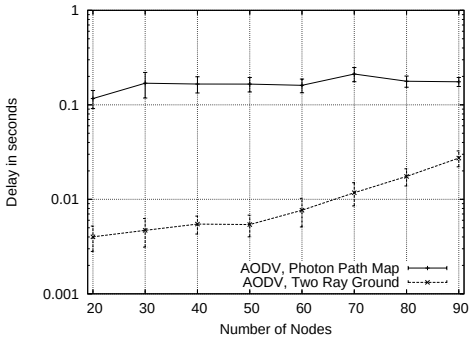


Figure 5: Packet delay computed by ns-2 for different number of nodes in a 802.11 based MANET with AODV routing. The figures computed with the Photon Path Map layer model the reality much better than with the Two Ray Ground approach, which is default in ns-2. The error bars show the 99% confidence interval.

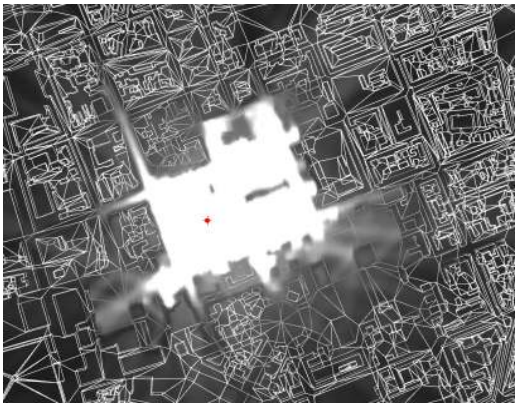


Figure 6: An excerpt of the volume image as it is produced by the Photon Path Map algorithm for the Munich dataset.

all positions, depending on the grid workload.

These maps are then used to do interpolated lookups of the field strength. The ns-2 generates requests for the field strength concerning a certain sender and receiver position. First we determine the k-nearest Photon Path Maps considering the transmitter position and visibility, as defined by obstacles in the scene. The selected maps are weighted by distance and the weighting coefficient is then used to do an interpolated lookup at the receiver position.

This can be done extremely fast and results in an approximated signal strength at the receiver position. This procedure results in a much more realistic result than the current two-ray-ground physical layer approach in ns-2 while it is only 1.5 to 1.7 times slower. Previous approaches using ray-tracing in the ns-2 were up to 100 times slower than with a simple propagation model [6]. As can be seen in Figure 5 using our physical layer model greatly improves on the overall ns-2 simulation run. With the simple physical model too low delay values are computed, and the network performance in the simulator is completely unrealistic.

4. EXPERIMENTAL RESULTS

We have run the algorithm mainly on three different scenarios for which real measurement data has been available. The first scene is the Munich dataset from the COST 231 project [5], which was originally sampled by Mannesmann Mobilfunk GmbH, Germany. This scene describes a microcell outdoor scenario of a few square-kilometers in size. Figure 6 depicts the volume image produced by our simulation and Figure 9 shows the measured and simulated field strengths as well as the error made in a plot.

One can see that spatially fast fluctuating reception, possibly due to shadowed areas, is difficult to model when far

from the transmitter. This is partly due to the fact that at this distance, the Photon Path Map is less densely sampled. Therefore some paths might be missed that contribute to the field strength at this point. Also quite a lot of the small scale changes in the measured field are probably due to small objects not represented in the geometry, or general dynamic scene effects. So the simulations do depend to some extend on accurate geometry data. One can not expect that with a rough model of the scene perfect simulations can be achieved. But even with missing details, like cars, trees or roof shape information, simulations of good quality can be computed.

As said before, one can specify different materials for each part of the buildings. If material properties like reflectance and translucency are not known, they can be computed by estimating them via some measurements in the scene and optimizing for these parameters in the simulation.

The other scenes are indoor scenarios, showing the rooms of two institutes. The first dataset has been provided by [1]. The field strength for this dataset has been sampled by using standard notebook computers and an architectural map of the building. Results are plotted in Figure 8(a) and the corresponding volume image is shown in Figure 7(a). The second indoor dataset has been provided [15]. The simulation results for this scenario can be seen in Figure 8(b) and the volume image in Figure 7(b).

The time complexity of the shooting step of the algorithm depends on the number of faces, the number of photons to be shot and the average length of a path. It scales $O(\log n_f)$ with the number of faces n_f, since a ray-triangle intersection and also a ray-diffraction edge detection in a kd-tree each take as long as a single search in that tree. The tree itself is built in a balanced way. The depth of the paths d can be considered constant, depending on the material values or an optional user defined cutoff. The number of photons n_p is also constant and user defined. So the shooting step of the algorithm takes $O(dn_p \log n_f)$ time, and as thus scales very well with large scenes that contain tens of thousands of triangles, although a higher number of photons is needed for scenes covering a larger area.

The following table shows the simulation results for the different scenes, including the mean error and the standard deviation from the measurements. All results were computed on a 2.2 GHz PC with 2 GByte of RAM.

Scene	Accuracy	Time	Mean error	Std. dev.
Munich	∼10 m	6m 53s	3.62 dB	4.68 dB
Indoor 1	∼0.2m	12.7s	5.1 dB	6.6 dB
Indoor 2	∼0.2m	15.8s	4.5 dB	5.6 dB

A comparison with other algorithms is somewhat difficult, due to the fact that there are few scenes with measurements publicly available which are used by most published works. Also several papers do not mention the used hardware, runtime, target complexity or an error measure.

In [19] a map of the city of Stuttgart is used, being 4km^2 in size. This is comparable to the 8km^2 of the Munich scene. For the Stuttgart data [19] give a preprocessing time of 31 minutes and a simulation time of 2 minutes for a small 0.64km^2 subset with 5 m resolution. They do not mention the hardware used, but considering the improvements that have been made in this area, and due to the fact that they only processed a small subset of the map, it makes sense to assume that for a scenario of 8km^2 their algorithm would still take between several minutes to half an hour on today's machines.

A direct comparison can be made with the Dominant Path Prediction algorithm for outdoor scenarios by [21]. Here, the Munich dataset has been used. In their paper they give the following values:

Method	Accuracy	Time	Mean error	Std. dev.
DPP	10 m	∼36s	3.7 dB	6.4 dB

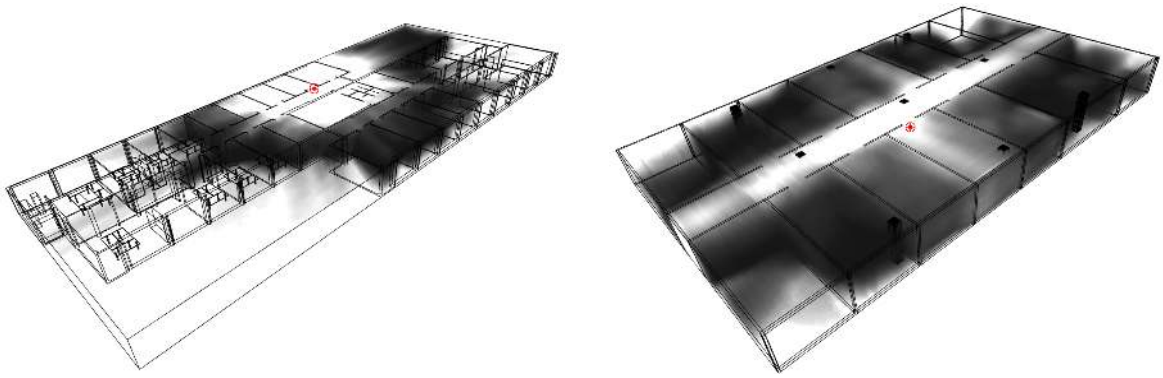
The mean error and the standard deviation is slightly worse compared to our algorithm. The performance gain of DPP can be explained due to the fact that it is a hybrid algorithm. It uses empirical path loss models as described in [26]. These models require very little computation time.

The advantage of our algorithm however is that it is much simpler, but also as accurate as the Dominant Path Prediction model. Furthermore it handles 3D scenes very well and processes almost arbitrary geometry in an efficient manner and produces extra data, like the delay spread information as described before. Since we showed that our approach can be done completely as a preprocessing step, the runtime will not affect a simulation in the ns-2.

5. CONCLUSION

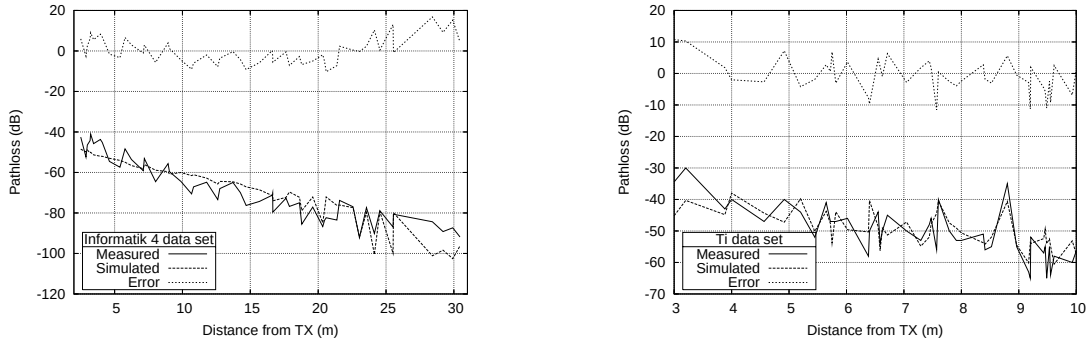
In this work we have presented a wave propagation algorithm with many advantages over existing algorithms. It is physically based and thus can be compared with real life measurements, an improvement of the scenario detail will result in a higher accuracy of the simulation. Standard triangular meshes can be used as input. These meshes can be easily produced for example from CAD-data for indoor scenarios or map data for outdoor scenarios. The algorithm handles indoor and outdoor cases equally well and does not need to be adapted to either case. It is even possible to use mixed scenarios, where urban microcell propagation is combined with, e.g., the reception of a mobile phone in a building.

Since the algorithm uses a Monte Carlo approach for the sampling of the ray-space, the target resolution of the algorithm has no great impact on the running time compared to other algorithms working on uniform grids. Also the final grid resolution does not reduce the accuracy of the raytracing step, as is the case with the approach by [19]. It only influences the accuracy of the kernel density estimator. The estimator only operates on voxels that contain energy, hence no time is spent on regions that never receive any signal. Furthermore with our algorithm one gets an increasingly more correct solution when increasing the number of rays. For a fast estimation of the field strength, a few thousand rays will do. For more accurate results one can spend as many time as needed on the ray shooting.



(a) (b)

Figure 7: Volume renderings for (a) the first indoor dataset and (b) the second indoor dataset.



(a) (b)

Figure 8: Experimental results for (a) the first indoor dataset and (b) the second indoor dataset. The dotted line shows the error, the dashed one shows the simulation and the solid curve denotes the measurements.

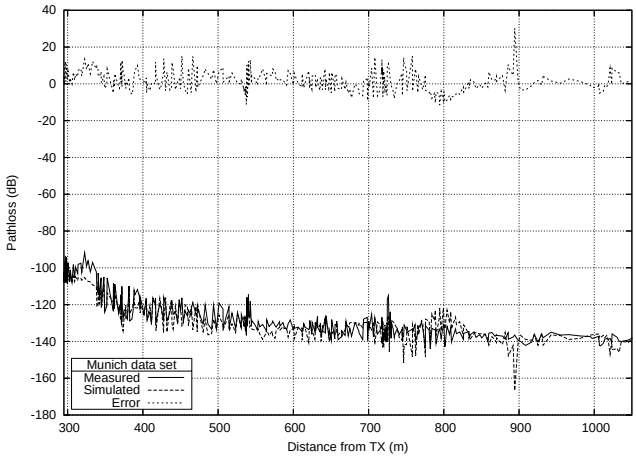


Figure 9: Simulation results for the Munich city center dataset. The lower two curves show the measured and simulated pathloss in dB and the curve at the top shows the error made in the simulation.

Another important advantage of this algorithm is that it produces a ray path-structure from which we derive information like the delay spread spectrum and the standard deviation of the delay spread. The paths include all possible combinations of reflection, transmission and diffraction. The path length is arbitrary and determined by the Russian Roulette procedure, which is unbiased. A biased manual recursion cutoff can be introduced to get faster but less accurate results.

6. REFERENCES

[1] Chair of Computer Science 4, RWTH Aachen University. http://www-i4.informatik.rwth-aachen.de/.

[2] H. Arslan and T. Yücek. Delay spread estimation for wireless communication systems. *iscc*, 0:282, 2003.

[3] J. Arvo and D. Kirk. Particle transport and image synthesis. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 63–66, New York, NY, USA, 1990. ACM Press.

[4] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, 1975.

[5] E. Damosso, editor. *Digital Mobile Radio: COST 231 View on the Evolution towards 3rd Generation Systems*. European Commission, 1998. Final Report of the COST 231 Project.

[6] J.-M. Dricot and P. D. Doncker. High-accuracy physical layer model for wireless network simulations in ns-2. In *Proceedings of the International Workshop on Wireless Ad-hoc Networks*, 2004.

[7] P. Dutré, P. Bekaert, and K. Bala. *Advanced Global Illumination*. A K Peters Ltd., 2003.

[8] P. Dutré, E. Lafortune, and Y. Willems. Monte Carlo light tracing with direct computation of pixel intensities. In *Proceedings of the 3rd International Conference on Computational Graphics and Visualisation Techniques, Alvor, P*, pages 128–137, Dec 1993.

[9] K. Fall and K. Varadhan. The ns-2 manual. Technical report, The VINT Project, UC Berkeley, LBL and Xerox PARC, 2003.

[10] E. R. Freniere, G. G. Gregory, and R. A. Hassler. Edge diffraction in monte carlo ray tracing. *Optical Design and Analysis Software*, 3780(1):151–157, 1999.

[11] D. S. Fussell and K. R. Subramanian. Fast ray tracing using K-d trees. Technical Report CS-TR-88-07, 1, 1988.

[12] H. W. Jensen and N. J. Christensen. Photon Maps in Bidirectional Monte Carlo Ray Tracing of Complex Objects. *Computers and Graphics*, 19:215–224, March 1995.

[13] J. B. Keller. Diffraction by an aperture. *Journal of Applied Physics*, 28(4):426–444, April 1957.

[14] R. G. Kouyoumjian and P. H. Pathak. A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. *Proceedings of IEEE*, 62:1448–1461, 1974.

[15] R. Mathar. Institute of Theoretical Information Technology, RWTH Aachen University. http://www.ti.rwth-aachen.de/.

[16] H. Moravec. 3D Graphics and the Wave Theory. In *Computer Graphics*, volume 15, pages 289–296, August 1981. presented at the 1981 Siggraph conference.

[17] R. J. Punnoose, P. V. Nikitin, and D. D. Stancil. Efficient simulation of ricean fading within a packet simulator. In *Vehicular Technology Conference*, September 2000.

[18] T. Rappaport. Characterization of uhf multipath radio channels in factory buildings. *Antennas and Propagation, IEEE Transactions on*, 37(8):1058–1069, Aug 1989.

[19] M. Schmeink and R. Mathar. Preprocessed indirect 3D-ray launching for urban microcell field strength prediction. In *AP 2000 Millennium Conference on Antennas and Propagation*, April 2000.

[20] H. Schober, F. Jondral, R. Stirling-Gallacher, and Z. Wang. Delay spread estimation for ofdm based mobile communication systems. In *Proceedings of the European Wireless Conference*, 2002.

[21] R. Wahl, G. Wölfle, P. Wertz, P. Wildbolz, and F. Landstorfer. Dominant path prediction model for urban scenarios. *14th IST Mobile and Wireless Communications Summit, Dresden (Germany)*, 2005.

[22] J. Walfisch and H. Bertoni. A theoretical model of UHF propagation in urban environments. *IEEE Transactions on Antennas and Propagation*, 36(12):1788–1796, December 1988.

[23] P. Wertz, R. Wahl, G. Wölfle, P. Wildbolz, and F. Landstorfer. Dominant path prediction model for indoor scenarios. *German Microwave Conference (GeMiC) 2005, University of Ulm*, 2005.

[24] G. Wölfle, B. Gschwendtner, and F. Landstorfer. Intelligent ray tracing - a new approach for field strength prediction in microcells. In *Vehicular Technology Conference*, volume 2, pages 790–794. IEEE, May 1997.

[25] G. Wölfle and F. Landstorfer. Field strength prediction in indoor environments with neural networks. In *Vehicular Technology Conference*, volume 1, pages 82–86. IEEE, 1997.

[26] G. Wölfle and F. Landstorfer. Dominant paths for the field strength prediction. In *Vehicular Technology Conference*, volume 1, pages 552–556. IEEE, 1998.