Field Strength Prediction for Environment Aware MIMO Channel Models

Florian Schröder, Michael Reyer, and Rudolf Mathar
Institute for Theoretical Information Technology
RWTH Aachen University
D-52074 Aachen, Germany
Email: {schroeder, reyer, mathar}@ti.rwth-aachen.de

***Abstract*—For planning, analysis, and optimization of modern, self organizing radio networks field strength prediction plays an essential role. A huge number of predictions is needed for planning purposes while high quality and level of detail are important as well. In this paper, we present a flexible ray optical approach for radio wave propagation to meet those demands. This approach allows easy exchange of the path loss model and produces multi path information enabling for environment aware MIMO channel models.**

I. Introduction

Fast radio wave propagation plays an essential role in planning, analysis, and optimization of modern radio networks. Therefore, detailed information, like multi path propagation, is needed and it should support MIMO techniques. The computational complexity is tremendous, but the overall computing time needs to be short for applicability. With respect to the application the ray optical approach is eligible.

The present work uses the following path loss model based on free space propagation. The model has been validated within previous work at our institute [1]. Using the overall distance $d(p)$ of the path p, neglecting the antenna gains, adapting the path loss exponent γ according to the environment, and introducing an estimator z_A mitigating imprecise information about transmit powers leads to

$$L_0^{dB}(p) = 20\lg(4\pi/\lambda) + z_A + 10\gamma \lg d(p)\,,$$

where λ denotes the wavelength. The impact of the effects is modeled by polynomial terms of a low degree denoted with k with the change of angle of each effect as input. This results in the overall attenuation of a path p

$$L^{dB}(p) = L_0^{dB}(p) + \sum_{x\in X}\sum_{i=1}^{n_x(p)}\sum_{j=0}^{k} z_{x,j}\alpha_{x,i}^{j}(p)$$

where $n_x(p)$ is the number of effects, $X = \{R, V, H\}$, $\alpha_{x,i}(p)$ are the changes of angle of the i-th occurrence of effect x, and $z_{x,j}$ are the estimated coefficients for the various effects. The shortcuts denote the physical effects R for reflection, V for vertical diffraction, and H for horizontal diffraction.

To achieve high performance we consider – from the hardware point of view – the following. 1. As hardware performance improvement has shifted from higher clock speed to more and more cores, parallel architectures need to be taken into account to enable for fast predictions. 2. Different parallel hardware architectures (CPU, GPU) can be utilized with the OpenCL [2] framework, which gains flexibility. 3. Plain shared memory multi core processors are easily exploited with the OpenMP API [3].

The input we require is 2.5D building data – building outline with height but no roof shapes – for urban environments. The environment data (buildings), shown in several figures, and measurement data is taken from the COST 231 project [4]. The following algorithm is made for calculating the relevant paths p in the urban environment efficiently and evaluate them with the given path loss model.

Section II addresses the interface of the presented prediction tool. Whereas Section III gives a detailed description of the core algorithm and physical effect handling (decision rules) as well as some concepts. Section IV presents an application, the ideas how MIMO is integrated and current investigations.

II. PIROPA

The presented algorithm for field strength prediction is called PIROPA, short hand for, *P*arallel *I*mplemented *R*ay *O*ptical *P*ropagation *A*lgorithm. It is a ray launcher designed for urban scenarios and implemented in `C++`. Since performance is a critical issue, strong effort is put into the parallel algorithm design. Data structures, for the unstructured mass of building input data, supporting the parallel design are determined in a preprocessing step. Considering its use case the program is constructed as a batch processing entity, reading input parameters from files and producing one or more output files. These files appear in form of *xml*, *csv*, *image* or plain line-serial *key-value* files, for further processing. Additionally, a graphical user interface – realized by the *Qt Ui* framework [5] – enables demonstration and debugging of the algorithm. It can visualize input and output files and allows limited interaction with the algorithm.

PIROPA is a deterministic algorithm, i. e., it reproduces results if it is started with the same input and parameters. The interface design allows for exchange of multi path information which enables interesting applications, e. g., the embedding in MIMO channel models. The application flow is as follows. First, the environmental information is preprocessed, i. e., meta information is generated and stored, some complex structures are simplified, and the data is partitioned. This has to be done only once per data set, and is for this reason outsourced in a separate program. The meta information, e. g., marking of adjacent walls or convex points[1], is used to improve speed and stability of the process. Furthermore, high level of detail in input data is reduced to a sufficient level for PIROPA, e. g., round structures are approximated by polygons with a low vertex count [6], [7]. Finally, a extended Binary Space Partitioning (BSP) is used for efficient data partitioning while enabling easy data access [8], [9]. The second most important task of the preprocessing is creating consistency within the environmental input data. Since there is no sole standard for data representation there are several types, e. g. shape files or map info files, which have no fixed interpretation for the physical units (cm, dm, m) or interpretation of the building height, height over ground vs. height over sea level, etc. The preprocessing produces a unified intermediate format. So certain terms can be guaranteed and extensive tests during runtime of PIROPA are omitted. Besides the building data a transmitter location, model parameters (see Section I), and receiver locations are required input parameters. The receivers are represented by a set of coordinates and may be anywhere as single points of receivers as well as placed as a whole plane with equidistant receivers. Receiver planes are used to generate images like shown in Fig. 2. The next section gives details on the core algorithm.

III. Algorithm and Interaction

After the algorithm is initialized, passing all the input data as described in Section II, the main program loop is started. Algorithm 1 outlines the main loop. It returns a tree structure of paths and effects. Its root node represents the transmitter. The tree nodes are incident locations where physical effect may occur. The branches of nodes describe rays shot from the node's position into different directions. Finally, the leaves represent receiver points which end a path. Overall, each path from root to leaf describes a ray path from the transmitter to a receiver point with its complete history of effects. Several paths may lead to the same receiver with different effects, which is utilized to describe the multi path environment for MIMO channel modeling.

[1]A convex point is a vertex v_i in a regular polygon, such that the triangle formed from v_{i-1}, v_i, v_{i+1} has a positive area.

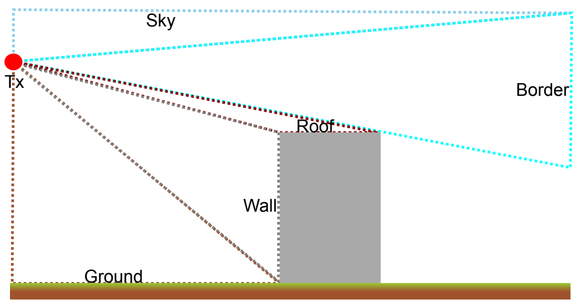

Fig. 1. Example for incident layout

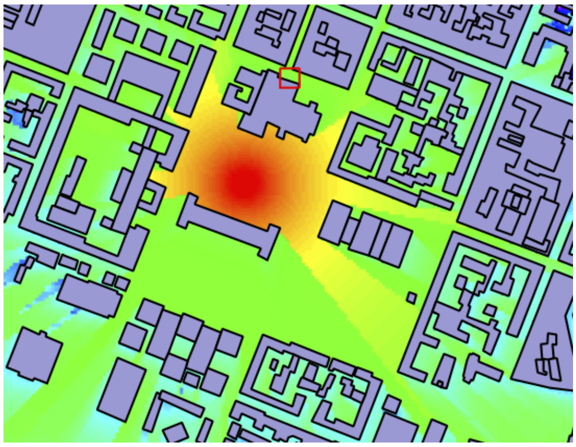

Fig. 2. A field strength prediction map

Algorithm 1 Outline of the main program loop

$finished \leftarrow$ **false**
$generation \leftarrow 0$
$process(RayGenerator)$
while not $finished$ **do**
  print $generation$
  $g \leftarrow process(GeometryManager)$
  $e \leftarrow process(EffectManager)$
  $v \leftarrow process(EvalManager)$
  $r \leftarrow process(RayGenerator)$
  $finished \leftarrow g \wedge e \wedge v \wedge r$
  $generation \leftarrow generation + 1$
end while

Algorithm 1 is initialized as not finished and starts with tree depth zero stored in $generation$. The first step in the while-loop is the so called *geometry manager* which will perform the intersection tests, locating incidents by intersecting the current bunch of brays with surfaces. The incidents calculated by the *geometry manager* are passed over to the *effect manager* which will decide on the type of incident. In the evaluation phase, executed by the *evaluation manager*, the path loss model is applied to the so far collected paths. Completely evaluated branches will be removed and stored in a condensed form in order to save memory. The last major step is the generation of new rays in the *ray generator*. This generation particularly depends on the type of incidents. If all four sub algorithms indicate that they are finished the loop will terminate, else the cycle will start with the next in size tree depth. Finally, the evaluation output is written into files. This might be a matrix with the strongest path for each receiver or a list of clusters (see Section IV) for each receiver point. To ensure the termination of the loop stop criteria which prevent further ray generation are implemented.

Parallelization enters this algorithm at several points. The geometric part is both one of the most computational intense parts of the algorithm and well suited for parallelization. A vast amount of identical calculations has to be performed on vectors of independent data sets and the greater part of the input data is constant. Each generation of rays depends only on its preceding generation such that parallel processing is enabled for each generation without data collision as long as the communication channels are synchronized. Consequently, the intersection algorithm is specialized for our use case, which means we do not use a full 3D intersection. We assume walls to be upright and that there is no free space below roofs. This holds true for the majority of relevant cases. Therefore, we split the full 3D intersection into two parts each of which is a 2D calculation similar to the suggestions in [10]. This method reduce the total number and complexity of calculations which improves the runtime significantly. The idea is to bundle rays shooting in the same horizontal direction (xy), but different vertical directions. Fig. 3 shows such a bundle

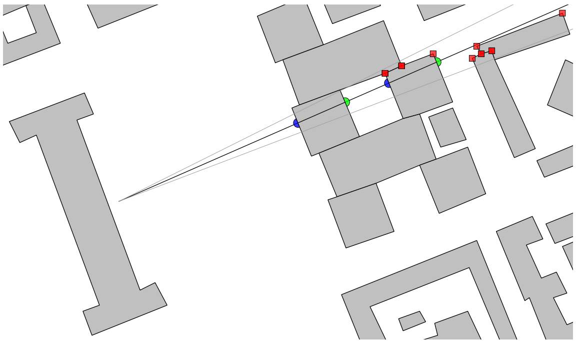

Fig. 3. Top view for a ray bundle in the (xy) plane

with entry and exit points of rays on buildings. In a second step, after the (xy) intersection, the ray will be distinguished in parts passing over the building, hitting the roof, and hitting the wall, see Fig. 1. These are the incidents, which have a location and a type (ground, wall, roof, ...). The effect manager from the algorithm consists of a set of rules (effect handler). The decision process determines physical effects based on the incident, the history of the path, and the hit surface. These rules are easy modifiable, such that modifying or replacing the propagation model does not need a complete redesign, but relatively small amount of coding. This is demonstrated in a

Algorithm 2 Example for simple reflection

if $incident.type = WALL$ **then**

$nn \leftarrow \mathbf{n}_{wall} \cdot \mathbf{n}_{wall}$

$dn \leftarrow \mathbf{n}_{wall} \cdot \mathbf{d}$

$k \leftarrow -2 * \left|\frac{dn}{nn}\right|$

$\mathbf{ww} \leftarrow \mathbf{n}_{wall} * k$

$\mathbf{r} \leftarrow \mathbf{d} - \mathbf{ww}$

$addTransmitter(\mathbf{o}, \mathbf{r})$

end if

trivial reflection example given in Algorithm 2. The decision is only based on the incident type (wall in this case) and calculates an exit angle for a new ray, applying the rule exit angle is equal to the incident angle. A new transmitter is created with only on ray at origin of the incident $\mathbf{o}$ in the new direction $\mathbf{r}$ and is enqueued for further processing. A more complex version could include scattering or take into account information on the hit surface. Other effects, e. g., diffraction effects, are handled analogous. First, it is decided whether the incident is handled within this effect. Handled incidents will be processed by treating this location as transmitter, generating new rays, and updating the history. It is permitted to apply different effects on one incident. For each incident the manager calls each incident handler. The flexibility of the framework allows for easy adaptation of the decision rules according to measurement based evaluations.

IV. Embedding into MIMO Channel Model

The obtained multi path information enables for combining ray launching and geometry based stochastic models [11]. Basically, we obtain many paths from the transmitter to a number of receiver points with respect to physical effects within the given environment. We gather amongst others the following information: 1. The angle of departure at the transmitter, 2. the angle of arrival at the receiver, 3. the distance, and 4. the attenuation of the path.

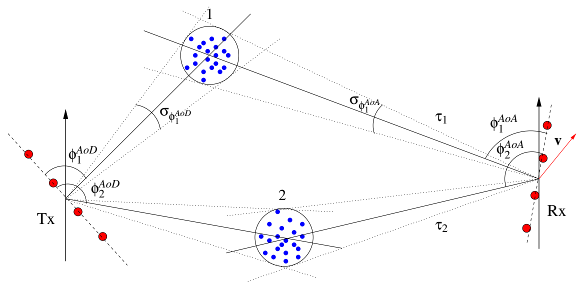

Fig. 4. MIMO multi path transmission with two clusters

The WINNER 2 MIMO Channel Model [12] uses clusters, see Fig. 4. They are generated randomly fulfilling some stochastically properties. Within each cluster there is a bunch of rays. As the required level of detail for those rays is higher than the environmental data allows for, our goal is to attain the clusters and generate the rays in the same way as in the WINNER 2 model. With our deterministic approach we provide an environment aware MIMO channel model. We have to determine if calculated paths belong to the same cluster. Fig. 5 depicts paths generated with PIROPA represented by their delay and attenuation. For a simple field strength map – the accompanying map is shown in Fig. 2 – the strongest one is taken, whereas for our application all of them are of relevance. We investigate different methods clustering the paths we obtain. Available information are the geometry of the paths including the history of physical effects. The key values are 1. the distance, giving the signal delay, 2. the attenuation, which is strongly influenced by physical effects, and 3. the angles at the transmitter and receiver. We need to develop an appropriate measure using these values. The data from Fig. 5 was processed in a simple clustering algorithm resulting in the data shown in Fig. 6. The number of clusters is quite high and suffices WINNER 2 model cluster input. However, the

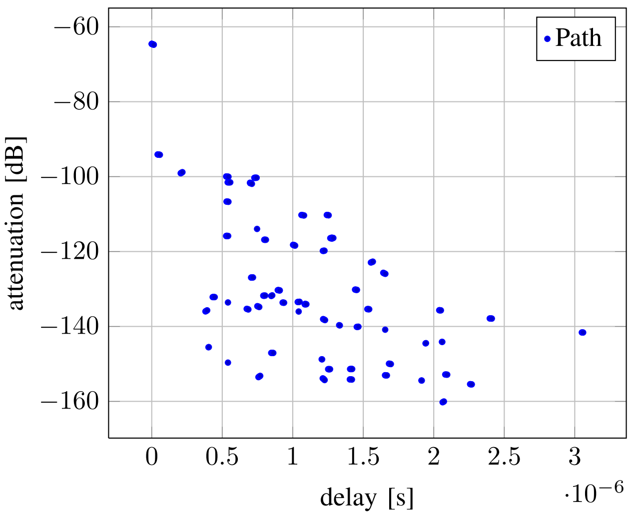

Fig. 5. Example of delay spread for paths generated by PIROPA

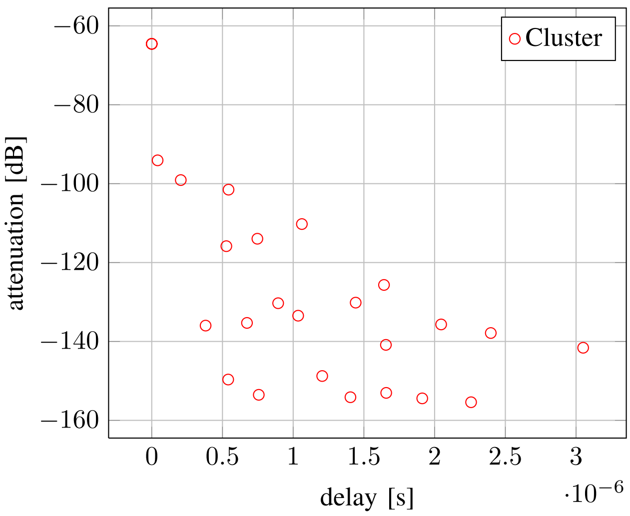

Fig. 6. Example of delay spread for clustered paths

evaluation of this model and the clustering method is subject of further research.

Fig. 7 shows a demonstrator GUI for the environment aware MIMO channel model. The left side depicts a 3D view of the building data including the field strength prediction from strong (red) to weak (green), the transmitter (light blue), and receiver points (pink). On the right side is a plot with the delay spread similar to Fig. 5 and the corresponding 2×2 MIMO channel matrix.

V. Conclusion

The contribution of the present work is a highly flexible ray launcher designed for parallel hardware enabling batch processing. A multitude of paths is calculated and stored efficiently with manageable resource consumption. The program is supporting current standard computing hardware as well as special high performance computing devices, e. g., Tesla GPUs. The output of PIROPA enables amongst others for calculating the delay spread at a receiver site and a MIMO channel matrix. The main advantage of this MIMO channel model is the environment awareness of local characteristics. In ongoing work we refine the detection of clusters. And finally an important step will be the comparison of the prediction results with measurement data.

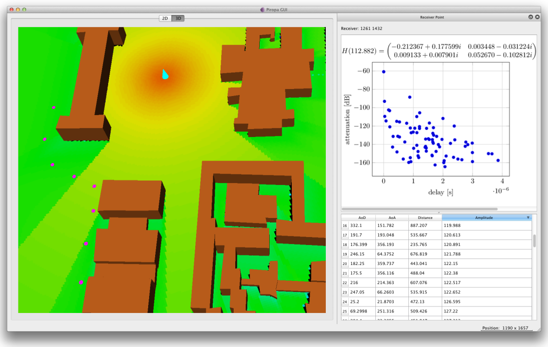

Fig. 7. Demonstrator GUI with a transmitter and multiple receivers

Acknowledgment

This research was partly supported by the UMIC excellence cluster of RWTH Aachen University.

References

[1] R. Mathar, M. Reyer, and M. Schmeink, “A cube oriented ray launching algorithm for 3D urban field strength prediction,” in *Proc. IEEE International Conference on Communications ICC ’07*, 2007, pp. 5034–5039.

[2] [Online]. Available: http://www.khronos.org/opencl/

[3] [Online]. Available: http://openmp.org/wp/

[4] “Cost 231 - urban micro cell measurements and building data.” [Online]. Available: http://www2.ihe.uni-karlsruhe.de/forschung/cost231/cost231.en.html

[5] [Online]. Available: http://qt.nokia.com/

[6] A. Aschrafi, P. Wertz, M. Layh, P. Landstorfer, G. Wolfle, and R. Wahl, “Impact of building database accuracy on predictions with wave propagation models in urban scenarios,” in *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, vol. 6, may 2006, pp. 2681 –2685.

[7] V. Degli-Esposti, F. Fuschini, E. Vitucci, and G. Falciasecca, “Speed-up techniques for ray tracing field prediction models,” *Antennas and Propagation, IEEE Transactions on*, vol. 57, no. 5, pp. 1469–1480, May 2009.

[8] F. Schröder, M. Reyer, and R. Mathar, “Efficient implementation and evaluation of parallel radio wave propagation,” in *EuCAP 2011 - 5th European Conference on Antennas and Propagation*, EUR Congressi in Rome, Italy, Apr. 2011, pp. 2466–2470. [Online]. Available: http://www.ti.rwth-aachen.de/publications/output.php?id=791&table=proceeding&type=pdf

[9] V. Havran, J. Bittner, and J. Zra, “Ray tracing with rope trees,” in *in Proceedings of 13th Spring Conference On Computer Graphics, Budmerice in Slovakia*, 1998, pp. 130–139.

[10] J.-P. Rossi and Y. Gabillet, “A mixed ray launching/tracing method for full 3-d uhf propagation modeling and comparison with wide-band measurements,” *Antennas and Propagation, IEEE Transactions on*, vol. 50, no. 4, pp. 517 –523, apr 2002.

[11] X. Xu, M. Reyer, F. Schröder, A. Engels, and R. Mathar, “A semi-stochastic radio propagation model for wireless MIMO channels,” in *IEEE International Symposium on Wireless Communication 2011*, Aachen, Deutschland, Nov. 2011, pp. 619–623. [Online]. Available: http://www.ti.rwth-aachen.de/publications/output.php?id=840&table=proceeding&type=pdf

[12] IST-4-027756 WINNER II, *D1.1.2 V1.2 WINNER II Channel Models*, Sep. 2007.