An Intelligent Ray Launching for Urban Prediction

Zhihua Lai, Nik Bessis, Guillaume de laRoche, Hui Song, Jie Zhang and Gordon Clapworthy
Institute for Research in Applicable Computing,
University of Bedfordshire,
Park Square, Luton LU1 3JU, United Kingdom

***Abstract*—With the increasing need of accuracy, deterministic propagation models have been widely adopted in site-specific wireless applications. Compared to empirical models, they are usually more accurate, but time-consuming. Several acceleration techniques have reduced the simulation time and yet the computation of typical scenarios remains complex. In this paper, a novel 3D model, namely IRLA (Intelligent Ray Launching Algorithm) is presented to predict a huge number of receiver pixels (few millions) in a large scenario (few square kilometers) within few seconds, which is beneficial to relevant academics and industries.**

I. Introduction

Propagation modelling plays an important role in wireless network planning and optimization. For instance, when planning the construction of a building, engineers need to know where to put the access points so that the signal coverage is optimal. Propagation modelling takes the building blueprint as input and outputs the potential optimal access point position and its signal strength coverage. In general, propagation models of interest should be sufficiently fast and accurate to meet commercial needs [2]. Several authors [1][3] have presented different categories of propagation models, which include pure empirical models, semi-empirical models, semi-deterministic models, deterministic models and hybrid models. An empirical model calculates the path loss based on distance, frequency of transmitter and mostly an empirical factor [1]. It does not consider the influence of obstacles. If the interactions of wave effects are high, the models will suffer from serious inaccuracy. However, empirical models such as COST HATA [1] do offer very fast predictions. Semi-empirical models [2] are empirical models with additional environmental information. For example, Multi-Wall model (MW) [3] takes into account the number of times that a wave goes through an obstacle (wall) between a transmitter and receiver. Semi-empirical models continue to suffered from low accuracy but provide better accuracy than purely empirical models. Semi-deterministic models are similar to semi-empirical models [3], but are based mainly on a deterministic part with an additional empirical estimation. They are quite popular as they offer a better accuracy than empirical and semi-empirical models while having low computation time. Deterministic models improve the accuracy by considering interactions of waves and are more time consuming, though several authors have proposed methods to accelerate the computation process and accuracy (See [4] etc). Existing deterministic methods include ray optical (ray tracing and ray launching) and Finite Difference Time Domain (FDTD) [5]. Several authors have proposed methods to accelerate the computation process and accuracy. Hybrid models usually combine more than one method. For example, ray tracing and FDTD are usually combined in outdoor to indoor scenarios [1].

Ray Optical methods include Ray Launching and Ray Tracing [6], which differ in the way in which they trace rays. Ray Launching emits the rays from the transmitter [8]. Signal strength degenerates as the rays propagate and additional loss is added when rays reflect or diffract from walls. The rays are separated by a small constant angle so distant area may not be covered when rays continue to diverge [7]. To resolve this, a smaller angle can be chosen but this leads to a much greater computation. Rays do not always hit exactly on an edge due to floating point errors - to solve this, a sphere structure is usually employed to catch the nearby rays [8]. Ray Tracing traces the rays backwards [8]. For example, when it mirrors the receiver R as R', by connecting R' with transmitter T, a intersection O is obtained. T - O - R is the exact reflected path between R and T. If there are few receiver locations, Ray Tracing might be enough to search a set of paths in between receivers and the transmitter. However, this approach slows down as the number of receivers grows.

For both Ray Launching and Ray Tracing, extensive ray paths have to be searched [7]. When rays hit the building edges, a huge number of secondary diffracted rays have to be considered, which exponentially increases the computation.

The rest of this paper is organized as follows. In the next section, a discrete ray launching model is presented, which provides high accuracy within a short computation time. After that, the propagation parameters used in the model are discussed. Calibration is discussed followed by the issues related to implementation. Results are compared to relevant models, followed by future work, which concludes the paper.

II. The Algorithms

Similar to CORLA (Cube Oriented Ray Launching Algorithm) [6], IRLA is based on discretized environment, that is, a specified scenario is rasterized into a huge number of cubes. Each cube is associated with a set of information. For example, a cube may be an integer which represents 32 kinds of boolean values. A cube may record whether it is a ground cube, or a wall cube.

Rasterization is the first step of the modeling, which has to be handled before the actual simulation starts. The simulation algorithm can be divided into three parts, namely, collecting

Line-Of-Sight cubes, intelligent vertical diffractions, intelligent horizontal diffractions.

Algorithm 1 Collecting L-O-S Cubes

```
C ← InitializeCubes()
B ← ∅
for all c ∈ C do
  if c_x = 0 or c_y = 0 or c_z = 0 or c_x = MaxX or c_y = MaxY or c_z = MaxZ then
    B ← B + c
  end if
end for/*Bordering Cubes are now stored in B*/
L ← ∅
T ← getTransmitterCube()
for all c in B do
  for all p in between T and c do
    if isFilled(p) then
      L ← L + p
      break
    end if
  end for
end for/*L-O-S Cubes are now stored in L*/
```

Algorithm 1 has complexity $O(n^2)$. It launches one ray to each bordering cube and puts the first filled cube into the L array if there is any. Finally, the algorithm collects all the cubes where secondary rays are needed to launch from. The running time of this part is trivial compared to the rest.

Algorithm 2 Intelligent Vertical Diffractions - 1

```
G ← ∅
for all c ∈ C do
  if isGround(c) and c_x = 0 or c_y = 0 or c_x = MaxX or c_y = MaxY then
    G ← G + c
  end if
end for/*Ground Bordering Cubes are now stored in G*/
for all c ∈ G do
  dealWithBuildingsInBetween(T, c)
end for
```

Algorithm 2 has complexity $O(n)$. It aims to find ground border cubes for the use in Algorithm 3. As it is assumed that buildings are located on one ground cube level, this procedure is extremely fast.

The Algorithm 3 scans each line from transmitter cube T to each ground border cube and obtains the number of vertical diffractions by jumping to next farthest 'visible' building. Two cubes are visible if there are no building cubes in between. This procedure has a complexity of $O(n^2)$ and can be accelerated by caching the 'visible' function. The algorithm is suitable for finding a roof top diffraction path for a huge number of pixels in an urban scenario in a short time. In an urban scenario, there are many obstacles and the signal strengths of distant pixels from the transmitter are usually

Algorithm 3 Intelligent Vertical Diffractions - 2

```
dealWithBuildingsInBetween(T, c)
  b ← ∅
  C ←getCubesFromTo(T, c)
  d ←NIL
  for all c ∈ C do
    if isGround(c) then
      if isGround(d) then
        if c_z > b(c)->z then
          b(c)->s ← c
          b(c)->e ← c /*update start and end positions*/
        else
          if c_h = b(c)->e_z then
            b(c)->e ← c /*update end position*/
          end if
        end if
      else
        new(b(c))
        b(c)->s ← c
        b(c)->e ← c /*create a new building block*/
      end if
    end if
    d ← c /*save current cube*/
    r ←getCubesWithXY(c_x, c_y) /*r is a array of cubes that x = c_x, y = c_y*/
    for all x ∈ r do
      i ← T
      while i ≠ x do
        j ←getCubesFromTo(i, c)
        for all y ∈ j do
          if isVisible(i, b(y)->s) then
            k ← y
          end if
        end for
        i ← k /*Move to farthest building it can see from cube i*/
        x_d ← x_d + 1 /*One more vertical diffraction*/
        if b(k)->s ≠ b(k)->e then
          x_d ← x_d + 1 /*One more diffraction*/
        end if
      end while
    end for
  end for
```

Algorithm 4 Intelligent Horizontal Diffractions & Reflections

```
for all c ∈ L do
  if isCubeReflected(c) then
    launchReflectedRayFrom(c); /*Reflected Rays*/
  end if
  if isCubeDiffracted(c) then
    launchDiffractedRayFrom(c); /*Diffracted Rays*/
  end if
end for
```

covered by such vertical diffraction. This algorithm guarantees to find at least a path between the transmitter and receiver. The Algorithm 4 has as input the secondary ray cubes from Algorithm 1. It scans each cube and launches reflected and diffracted rays, respectively. It checks secondary ray cubes of the iteration and recursively iterates until either a maximum number of iterations or a threshold of maximum path loss is reached.

III. Model for Radiowave Propagation

The model to calculate Free Space Path Loss [1] is given in Equation 1

$$L_{\text{FSPL}} = \frac{P_t}{P_r} = \frac{P_t(\frac{4\pi d}{\lambda})^2}{G_t G_r} \tag{1}$$

where
d is distance between emitter and receiver;
f is the frequency;
P_t is the estimated transmitter power;
P_r is the estimated receiver power;
G_t is the gain of transmitter antenna;
G_r is the gain of receiver antenna.
To calculate path loss from transmitter to receiver, given the multi-path information, Equation (2) can be used.

$$\text{PL} = L_{\text{Path}} + L_{\text{Diffraction}} + L_{\text{Transmission}} + L_{\text{Reflection}} \tag{2}$$

where
$L_{\text{Path}} = \sum_{i=1}^{N_p} \text{FSPL}(d_i, f)$
$L_{\text{Diffraction}} = N_d(L_d D(\alpha))$
$L_{\text{Transmission}} = N_t(L_t)$
$L_{\text{Reflection}} = N_r(L_r R(\alpha))$
In the IRLA model, diffraction, transmission, reflection are considered, with the losses defined in L_d, L_t, L_r, respectively. L_{Path} sums up the loss from different path segments. The $L_{\text{Diffraction}}$ evaluates diffraction loss for each diffraction path. Due to lack of material information, a simplified function can be evaluated by considering the path length and direction change angle α from the incident ray to the diffracted ray. $L_{\text{Transmission}}$ evaluates an empirical transmission loss of building walls that is based on the number of transmitted walls and a constant factor. $L_{\text{Reflection}}$ evaluates reflection loss for each reflection path. The additional losses due to direction angle α change for reflection and diffraction are modeled in $R(\alpha)$ and $D(\alpha)$ respectively. Diffraction loss $D(\alpha)$ is calculated by employing Uniformed Theory of Diffraction (UTD) [9]. The path loss obtained in equation 2 assumes the use of an omni-directional antenna pattern. To include the antenna pattern in the model, an initial antenna gain is set at the beginning of each ray emitted from transmitter. Acceleration can be obtained by roughly including the antenna pattern adjustment when the path loss calculation is finished.

IV. Parameter Calibration

Due to limited environment database information and unpredictable phenomena [6] such as moving vehicles or fast fading, a calibration process is usually needed to set the propagation model parameters. Measurement data is used to make simulation result fit to reality. In IRLA, simulated annealing is used. Parameters included in the calibration are L_d, L_t, L_r, the free space path loss coefficient and the material coefficient for outdoor concrete. As all multipath information of pixels is stored in memory after one simulation, the calibration process is fast.

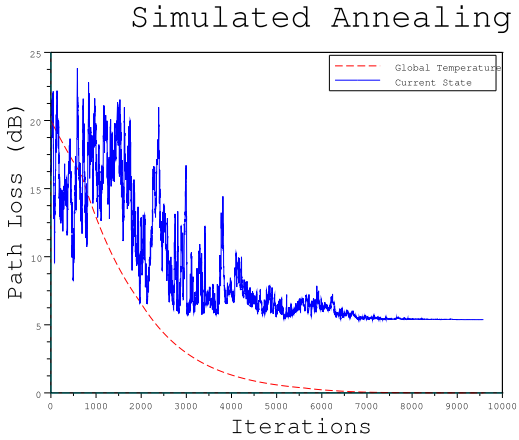


Fig. 1. Calibration Based on Simulated Annealing

Each measurement point M_i (with $i \in [1, N_p]$, N_p is the number of points) is assigned with a measurement path loss. For each iteration of calibration process, P_i (with $i \in [1, N_p]$, N_p is the number of points) is assigned with a prediction path loss. Thus, the error can be defined as

$$E_i = M_i - P_i \tag{3}$$

Mean Error (ME) between measurement and prediction can be also obtained by $ME = \frac{\sum_{i=1}^{N_p} E_i}{N_p}$. ME can be interpreted as the offset between measurement and prediction, and therefore, ME can be added to predictions, which can greatly reduce the errors. Root Mean Square Error (RMSE) is often used as a estimate of accuracy. RMSE will be treated as the cost function to minimize during calibration. It can be calculated as

$$\text{RMSE} = \sqrt{\frac{1}{N_p}\sum_{i=1}^{N_p} E_i^2} \tag{4}$$

The parameters are all real, which yields an infinite search space. In this work, Simulated Annealing (SA), a meta-heuristics-based optimization technique is chosen to calibrate IRLA. Each iteration of SA obtains a RMSE, and based on the probability and current temperature, a new state is generated; finally, it converges to an optimal solution (Figure 1). The calibration stops when a small RMSE is obtained, the global temperature is too low, or a maximum step is reached.

V. Implementation

The Object Pascal implementation utilizes IRLA with efficiency while a relatively high accuracy is maintained. To speed up IRLA, the following approaches can be taken.

- Database Simplification: Due to the lack of scenario information, the outdoor building data is usually simplified by defining a number of horizontal polygons and their height information (2.5D) (Figure 2). The material of each building is considered to be concrete. The discretizion process should output cubes with different kind of geometry data, for example, building edges, corners, roofs, inner, etc.

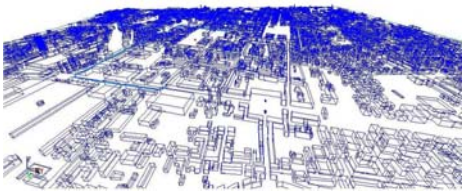


Fig. 2. 2.5D Building Data of COST Munich

- Double Marking (DM): DM occurs when duplicated rays passing though a cube are counted more than once. Avoiding DM is important for speed and accuracy.
  - Single Rays DM: SRDM refers to the situation in which a cube is marked more than once when it comes from a single ray emitter. L-O-S and reflection sources emit one ray at a time. A cube can only be marked once from the same single ray emitter source.
  - Multiple Rays DM: MRDM occurs when a cube is marked more than once and it comes from a multi-source ray emitter. For example, a diffraction cube emits several rays at a time. A cube can only be marked once from the same multi-source ray emitter source.
- Multi-threading: IRLA has employed multi-threading to reduce the running time. To avoid thread conflict as much as possible, rays launched by threads at the same time should be widely separated (i.e. the angles between launching directions are above a threshold value).

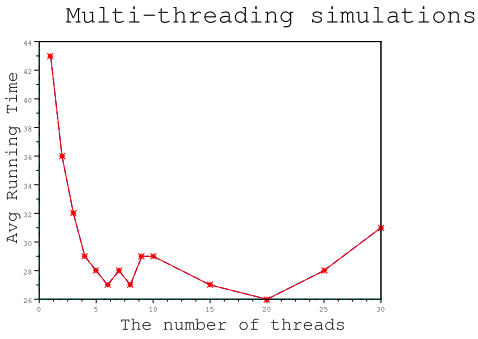


Fig. 3. Multi Threading Improvement on AMD64 Dual

From Figure 3, considerable time savings can be made by employing multi-threading technology. Performance improves dramatically when a few threads are used, compared to a single thread, but adding further threads produces little further gain,

TABLE I
RUNNING TIME OF IRLA

CPU	Memory	Computation(s)	Calibration(s)
AMD64 Dual, 2 X 2.6GHz	3.25GB	19	59
AMD2600+, 1 X 1.9GHz	756MB	27	120

TABLE II
NETWORK CONFIGURATIONS

Area	8.1 km^2 X 100m
Resolution	5 X 5 X 5
Maximum Reflection	3
Maximum Horizontal Diffraction	7
Maximum Vertical Diffraction	**Unlimited** *
Maximum Transmission	**Unlimited** *
Tuning Iterations	10, 000 **

* until signal strength is under threshold
** until temperature falls below threshold

which is due to the limit of CPU and the threading delay caused by resource competitions.

VI. RESULTS

IRLA provides a fast and accurate prediction of an urban scenario. The coverage prediction for COST-Munich [10] is shown in Figure 4.

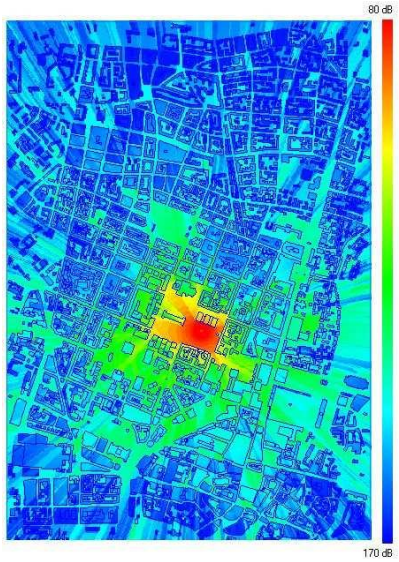


Fig. 4. Munich Signal Strength Prediction, $8.1km^2$

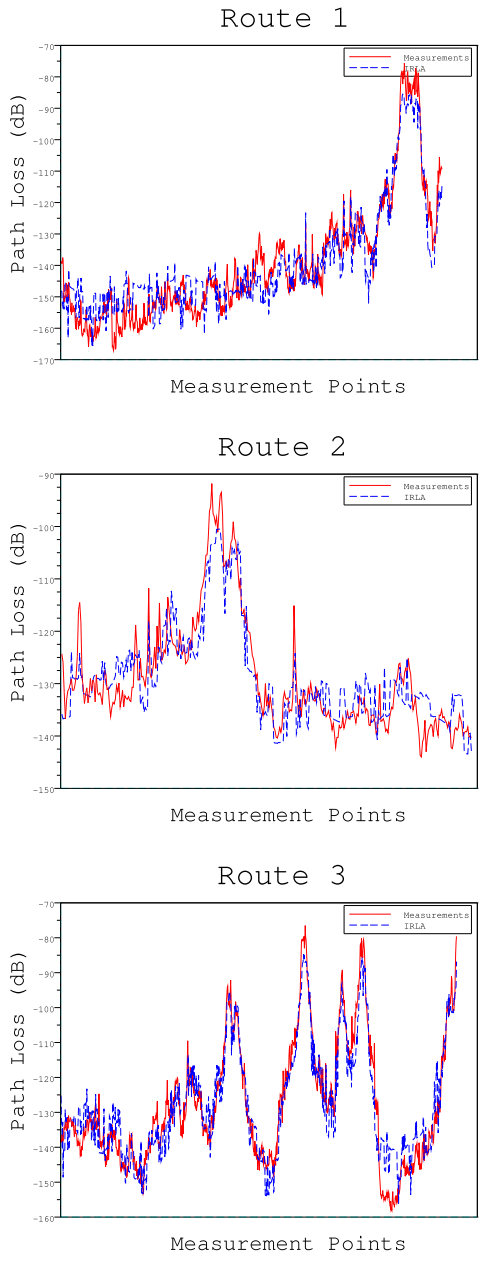
IRLA is intended to obtain a huge number of pixels, not just the ground-level pixels. Results show that around 6.5 million pixels are obtained in around 20 seconds. For the COST-Munich scenario, measurement route 1 is used to calibrate. The running time has been listed in Table I, which shows a relatively fast approach on standard PC. Table II gives the network parameters for running IRLA on COST Munich scenario. A large number of ray iterations allows IRLA to simulate most of the ray propagation phenomena in urban environment. In Table III and Figure 5, accuracy of results obtained are listed and compared. IRLA tends to give a very

TABLE III
ACCURACY OF IRLA ON COST-MUNICH

Routes	STD	RMSE	Mean Error	Correlation
1	6.889	6.884	0.001	93.315
2	4.950	4.914	-0.011	88.767
3	6.035	6.021	-0.006	94.153

fast prediction of a huge number of pixels within a short amount of time, and keep the accuracy at a reasonable level.

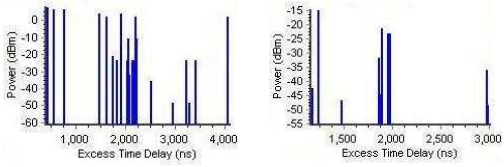
Fig. 5. Route 1, 2, and 3 of COST-Munich, Simulation and Measurement (Red Line)



VII. CONCLUSIONS AND FUTURE WORK

This paper has presented IRLA, which can be used to perform fast and accurate signal coverage prediction of urban scenarios. Results have been shown, which shows that IRLA is suitable in wireless network planning. As IRLA has an inherently parallel structure (i.e. rays can be handled in parallel), using parallel and distributed computing technologies is likely to improve the performance. Applications of using IRLA in wireless network planning and optimization are to be studied. For example, IRLA can be used to predict fading phenomena by extracting multipath information. Sample can be seen in Figure 6 where multipaths of a LOS and a N-LOS cases are extracted, respectively.

Fig. 6. Power Delay Profile for LOS, N-LOS



ACKNOWLEDGEMENTS

This work is supported by the EU-FP6 "RANPLAN-HEC" project under grant number MEST-CT-2005-020958 and "GAWIND" under grant number MTKD-CT-2006-042783. Special thanks have to be given to Dr Michael Reyer from RWTH Aachen University for his generous help, without which IRLA would have not been possible.

REFERENCES

[1] T. Kurner, "Radio wave propagation part one 'theoretical aspects'," in *1st COST 2100 Training School*, Wroclaw, Poland, 2 2008.

[2] Y. Lostanlen, "Radio wave propagation part two 'practical aspects'," in *1st COST 2100 Training School*, Wroclaw, Poland, 2 2008.

[3] M. Klepal, "Novel approach to indoor electromagnetic wave propagation modeling," Ph.D. dissertation, Czech Technical University in Prague, 2003.

[4] N. A. Carr, J. Hoberock, K. Crane, and J. Hart, "Fast gpu ray tracing of dynamic meshes using geometry images," in *Proceedings of Graphics Interface*, Quebec, Canada, 6 2006, pp. 203–209.

[5] A. Valcarce, G. D. L. Roche, and J. Zhang, "On the use of a lower frequency in finite difference simulations for urban radio coverage," in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, Marina Bay, Singapore, 5 2008, pp. 270–274.

[6] R. Mathar, M. Reyer, and M. Schmeink, "A cube oriented ray launching algorithm for 3d urban field strength prediction," in *IEEE Communications Society*, vol. 49, 6 2007, pp. 5034–5039.

[7] B. Gschwendtner, G. Wolfle, B. Burk, and F. Landstorfer, "Ray tracing vs. ray launching in 3-d microcell modelling," in *European Personal and Mobile Communications Conference EPMCC95*, Bologna, Italy, 10 1995, pp. 74–79.

[8] R. Hoppe, G. Wolfle, and F. Landstorfer, "Accelerated ray optical propagation modeling for the planning of wireless communication networks," in *IEEE Radio and Wireless Conference, RAWCON 99*, 1999. [Online]. Available: citeseer.ist.psu.edu/433917.html

[9] D. McNamara, C. Pistorius, and J. Malherbe, *Introduction to the uniform geometrical theory of diffraction*. Artech House Publishers, 1 1990.

[10] "COST 231 - urban micro cell measurements and building data," http://www2.ihe.uni-karlsruhe.de/forschung/cost231/cost231.en.html.