Full-wave Simulations of Electromagnetic Scattering Problems with Billions of Unknowns

Bart Michiels, Jan Fostier, *Senior Member, IEEE,*
Ignace Bogaert, *Member, IEEE,*
and Daniël De Zutter, *Fellow, IEEE*

***Abstract*—Algorithmic improvements to the parallel, distributed-memory Multilevel Fast Multipole Algorithm (MLFMA) have resulted in implementations with favorable weak scaling properties. This allows for the simulation of increasingly larger electromagnetic problems, provided that sufficient computational resources are available. This is demonstrated by presenting the full-wave simulations of extremely large perfectly electrically conducting (PEC) sphere and Thunderbird geometries. Both problems are formulated using the combined field integral equation (CFIE) and discretized in over respectively 3 and 2.5 billion unknowns. They are solved using 4 096 CPU cores and 25 TByte of memory. To the best of our knowledge, this is the largest number of unknowns and the highest amount of parallel processes reported to date, for this type of simulation. Additionally, it is demonstrated that the implementation attains a high parallel speedup and efficiency.**

***Index Terms*—MLFMA, Method of Moments (MoM), parallel computing, distributed-memory**

I. Introduction

The Multilevel Fast Multipole Algorithm (MLFMA) is one the most popular methods to accelerate the matrix-vector multiplication during the iterative Method of Moments (MoM) solution of electromagnetic scattering problems formulated by means of boundary integral equations (BIE). The MLFMA reduces the computational complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$, with N the number of unknowns. For an introduction to the MLFMA, we refer the reader to [1]. State-of-the-art MLFMA implementations can handle several millions of unknowns on a single workstation. However, electrically large scattering problems (i.e. geometry size $\gg$ wavelength λ) might require a discretization into hundreds of millions, if not billions, of unknowns. Therefore, significant efforts have been devoted to the development of distributed-memory parallel MLFMA implementations that can make efficient use of large computational clusters [2]–[17].

Baseline approaches to the parallel, distributed-memory MLFMA rely on *spatial* partitioning, i.e., the distribution of boxes in the MLFMA tree among the computational nodes [2]–[5]. Such implementations fail to properly distribute the workload at higher levels of the MLFMA tree and do not scale well beyond a few tens of nodes. In [6], [7], this bottleneck was partly addressed by a *hybrid* approach: spatial partitioning is used for the lower levels of the MLFMA tree whereas *k*-space partitioning is used at the top levels. Rather than distributing boxes, *k*-space partitioning relies on the distribution of the radiation pattern sampling points among nodes. Further improvements were proposed in [8], [9], by means of a *hierarchical* partitioning scheme that allows for a gradual transition between spatial and *k*-space partitioning.

In [14], the three existing schemes (spatial, hybrid and hierarchical) were found to exhibit unfavorable weak scaling behavior. In a weak scaling analysis, both the problem size N and the number of parallel processes P are increased proportionally, keeping the problem size per process N/P constant. In [14], it was found that at least one of the processes has a per-level computational complexity of $\mathcal{O}(N)$ when using spatial partitioning and $\mathcal{O}(\sqrt{N})$ when using the hybrid or hierarchical scheme. This means that these schemes fail to distribute the workload evenly among nodes for a sufficiently high N and P. This load imbalance becomes even more stringent when N and P are increased further, putting a limit on the problem size N that can be handled in practice.

In [14], [15], we proposed a modification to the hierarchical scheme (called *B-HiP* – Blockwise Hierarchical Partitioning) with a computational complexity of $\mathcal{O}(\log N)$ per process. Whereas the 'original' hierarchical partitioning scheme [8], [9] relies on a distribution of radiation pattern samples in one angular direction (e.g. elevation), the proposed algorithm relies on a two-dimensional partitioning of the radiation pattern samples in both elevation and azimuth. As elaborated on in [14], this two-dimensional partitioning overcomes a bottleneck in the partitioning of radiation patterns that becomes apparent only for very large problem sizes and a very high number of CPU cores. As the current trend is to incorporate more and more parallelism to advance compute power, such computational architectures will become increasingly widespread.

In this Communication, we demonstrate that an implementation of the B-HiP algorithm is indeed able to scale beyond thousands of CPU cores by solving two problems with respectively 3 and 2.5 billion of unknowns, using 4096 CPU cores. To the best of our knowledge, this is the highest number of parallel processes and the largest number of unknowns reported for the high-frequency MLFMA. Additionally, we demonstrate that the implementation attains a very high parallel speedup and efficiency

II. Results

The simulations are performed on a Tier 1 cluster consisting of 512 nodes, each node containing two 8-core Intel Xeon E5-2670 processors and 64 GByte of RAM. This amounts to a total of 8 192 CPU cores and 32 TByte of RAM. The nodes are interconnected by an FDR Infiniband network (fat tree topology with a 1:2 oversubscription).

The parallel MLFMA solver is implemented in C/C++. The inter-process communication is handled by the Message Passing Interface (MPI). All numerical computations were performed in single-precision. In our implementation, the number of radiation pattern partitions is a power of four (i.e., 1, 4, 16, 64, ...). Hence, the number of processes P is also required to be a power of four. This means that only 4 096 CPU cores on this cluster can be used.

A. Parallel Speedup and Efficiency

We consider the scattering of a plane wave by a 40.03λ perfectly electrically conducting (PEC) sphere. The problem is

The authors are with the Department of Information Technology (INTEC), Ghent University, Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium (e-mail: jan.fostier@intec.ugent.be).

Manuscript received January 3rd, 2014; revised October 25th, 2014.

TABLE I
RUNTIME t_P, SPEEDUP S_P AND EFFICIENCY η_P FOR A SINGLE TFQMR ITERATION AS A FUNCTION OF AN INCREASING NUMBER OF PROCESSES P AND A FIXED-SIZE PROBLEM ($N = 6\,027\,555$).

P	1	4	16	64	256	1024	4 096
t_P (s)	3 579	1 148	303.5	76.58	18.80	5.131	1.589
S_P	1.00	3.12	11.79	46.73	190.4	696.0	2 252
η_P (%)	100	77.93	73.70	73.01	74.36	67.97	54.98

TABLE II
SIMULATION DETAILS

	Sphere	Thunderbird
Object size	$1\,801.25\,\lambda$	$2\,695.43\,\lambda$
Total number of unknowns N	3 053 598 633	2 506 261 716
RWG discretization	$\lambda/10$	$\lambda/10$
Integral Equation (IE)	CFIE, $\alpha = 0.5$	CFIE, $\alpha = 0.5$
Total number of CPU cores P	4 096	4 096
Total memory usage	24.9 TByte	24.2 TByte
Number of MLFMA-levels	15	15
Minimal box size	$0.2\,\lambda$	$0.2\,\lambda$
Number of multipoles L	$8, 10, \ldots, 35\,780$	$10, 11, \ldots, 35\,781$
Krylov method	TFQMR	TFQMR
Block-Jacobi preconditioner	$0.2\lambda \times 0.2\lambda$	$0.2\lambda \times 0.2\lambda$
MLFMA precision	10^{-2}	10^{-2}
Number of iterations	150	150
Relative residual norm	0.0238	0.0387
Matrix-vector product time	4m 53s	4m 8s

discretized into $6\,027\,555$ unknowns and is sufficiently small to be handled by a single computing node. This problem is solved repeatedly for an increasing number of parallel processes ($P = 1, 4, 16, \ldots, 4\,096$) and the time for a TFQMR iteration t_P is measured accordingly (averaged over a number of iterations). The speedup S_P and the parallel efficiency η_P with respect to the computations on a single core are defined as

$$S_P = \frac{t_1}{t_P} \tag{1}$$

and

$$\eta_P = \frac{S_P}{P} \tag{2}$$

respectively.

In the ideal case, the speedup is equal to the number of processes used (i.e., $S_P = P$) and the efficiency $\eta_P = 1$ (100%). However, from Amdahl's law [18] it follows that $\eta_P \to 0$ when $P \to +\infty$ for any parallel algorithm, as the unavoidable sequential part prevents t_P to decrease below a certain threshold.

Table I lists the time t_P per TFQMR iteration, the speedup S_P and the parallel efficiency η_P as a function of P. Note that a single TFQMR iteration involves two matrix-vector products in order to compute a new solution vector and a third matrix-vector product to compute the relative residual norm. For $P = 1$ and $P = 4$, a single node was used with the remaining cores left idle. For all other cases, nodes were filled to full capacity, i.e., 16 processes per node. Using 4096CPU cores, a speedup of 2 252 is obtained compared to the sequential case (P = 1), reducing t_P from one hour to only one and a half seconds. This corresponds to an efficiency of $\eta = 55\%$. Using $P = 4\,096$ processes, the average number of unknowns per process (i.e., N/P) is only about 1 500. Such low N/P ratios make it more difficult to achieve an evenly balanced load,

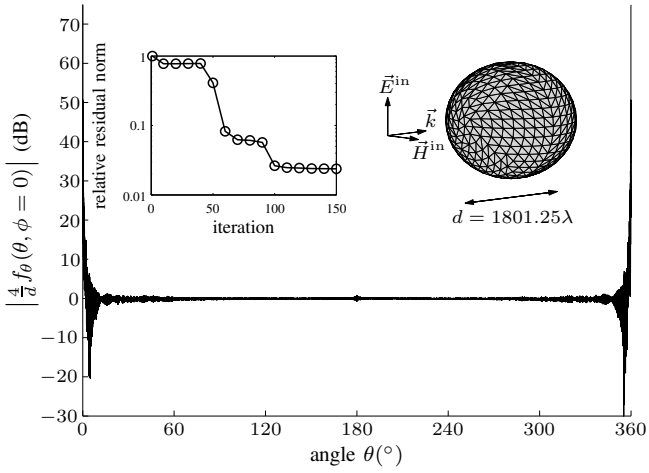

Fig. 1. The normalized radiation pattern for a PEC sphere with a diameter $d = 1801.25\lambda$. Inset: convergence behavior over 150 iterations.

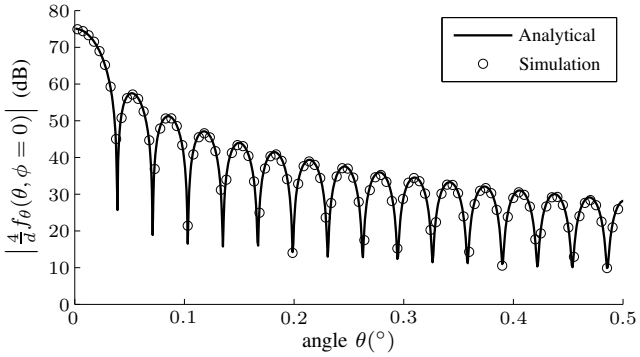

Fig. 2. A detailed view of the normalized radiation pattern for a PEC sphere with a diameter $d = 1801.25\lambda$ around the forward scattering direction.

explaining the loss of efficiency. Nevertheless, the observed parallel efficiencies are among the highest reported in literature and for larger problems, such as the ones presented below, even higher efficiencies can be expected.

B. Extremely large-scale simulation of a sphere

We consider the scattering of a plane wave by an extremely large sphere, with a diameter of 1801.25λ, discretized into more than three billion unknowns and formulated using the Combined Field Integral Equation (CFIE) [20] with a combination factor $\alpha = 0.5$. For each of the 512 nodes in the cluster, 8 CPU cores were used (4 096 in total) in order to be able to employ the full memory capacity of the 512 nodes. The MLFMA tree consists of 15 levels. At the four lowest levels of the tree, spatial partitioning is used. From the 5^{th} to the 10^{th} level, the radiation pattern sampling points are distributed among an increasing number of $4, 16, \ldots, 4\,096$ partitions until full *k*-space partitioning is obtained at the 10^{th} and higher levels. This corresponds to the hierarchical

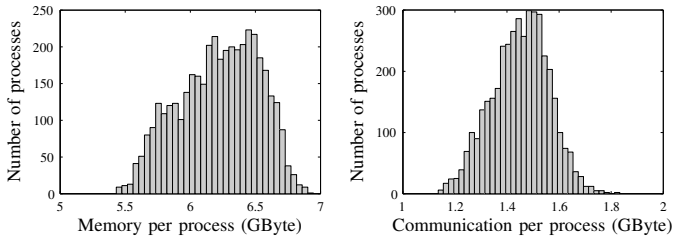

Fig. 3. Histogram of the memory usage per process (left) and the total communication volume (send + receive) per process (right) for the 'Thunderbird' geometry.

partitioning scheme, however, crucial to the B-HiP scheme is that the radiation patterns are distributed in both elevation and azimuth. A block-Jacobi preconditioner was used, with diagonal blocks corresponding to the self-interactions of the lowest level boxes of the MLFMA tree. After 150 iterations, a relative residual error of 0.0238 was obtained, using the TFQMR iterative method. The total solution time was composed of 6 hours and 40 minutes of setup time, 38 hours and 28 minutes of solving and 2 minutes to compute the output radiation pattern. The latter was computed recursively, similar to the aggregation phase in the MLFMA. In total, 25 TByte RAM was used. Additional simulation details are presented in Table II. Parameters were chosen such that memory use is minimized, rather than runtime.

Fig. 1 displays the absolute value of $\frac{4}{d} f_\theta(\theta, \phi = 0)$, the θ-component of the normalized radiation pattern in the $\phi = 0$ plane for the full $\theta = 0^\circ \ldots 360^\circ$ range, with d the diameter of the sphere. Fig. 2 shows the forward scattering direction for $\theta = 0^\circ \ldots 0.5^\circ$. The result of the simulation corresponds very well to the analytical Mie solution [19], with a root mean square (RMS) error of 0.568 dB. Here, the RMS error is defined as

$$\sqrt{\frac{1}{N} \sum_{i=1}^{N} \left| \frac{4}{d} f_\theta(\theta, \phi = 0)_{\text{sim}} - \frac{4}{d} f_\theta(\theta, \phi = 0)_{\text{anal}} \right|^2} \qquad (3)$$

with $\frac{4}{d} f_\theta(\theta, \phi = 0)$ the normalized radiation pattern evaluated in $N = 71\,564$ equidistant directions $\theta_i \in [0^\circ \ldots 360^\circ]$ and expressed in dB.

The workload for the simulation was partitioned such that each process was attributed (almost) the same number of unknowns, namely $745\,507 \pm 10$ unknowns. Fig. 3 (left) displays a histogram representing the distribution of the memory usage per process among all $4\,096$ processes. This includes all required memory for geometry, near interactions and (dis)aggregation matrices, radiation patterns, translation operators and communication buffers. The memory usage per process ranges from a minimum of 5.43 GByte to a maximum 6.95 GByte. On average, a process required 6.22 GByte of memory, which corresponds to a total of 24.9 TByte for the entire simulation. Similarly, Fig. 3 (right) displays the distribution of the total communication volume per process (both sending and receiving). On average, a process has to send or receive 1.45 GByte of data. In the B-HiP algorithm, both

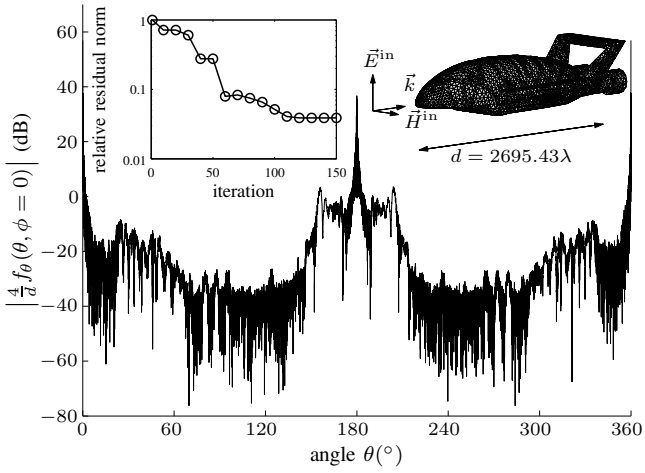

Fig. 4. The normalized radiation pattern for a PEC 'Thunderbird' discretized in over 2.5 billion unknowns. Inset: convergence behavior over 150 iterations.

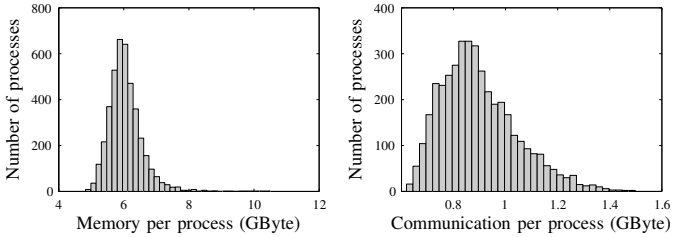

Fig. 5. Histogram of the memory usage per process (left) and the total communication volume (send + receive) per process (right) for the sphere geometry.

memory usage and communication volume are guaranteed to increase only logarithmically when further increasing N and P. The size of the problems that can be handled with the current implementation is limited by the available memory on the cluster.

C. Extremely large-scale simulation of a Thunderbird

In order demonstrate the ability of the method to handle arbitrary geometries we consider a second example in the form of a 'Thunderbird' geometry. This model was created using GID and the initial mesh contained $11\,214$ faces. By applying recursive subdivision of the triangles, a mesh with over 2.5 billion unknowns was obtained.

The problem was again solved using 4096 parallel processes and identical parameters as for the sphere, with the exception of a slightly higher multipole truncation number L (see Table II). The latter was required to ensure the accurate evaluation of FMM interactions even when RWGs protrude partly from the box in which they are contained. After 150 iterations, a relative residual error of 0.0387 was obtained. Fig. 4 depicts the radiation pattern and convergence behavior. Finally, Fig. 5 shows the histogram of the memory requirements and communication volumes per process. Note that for this geometry,

the memory distribution is somewhat wider compared to that of the sphere. This is due to a higher variation in the memory requirements to store the near interactions, which is in turn a consequence of the mesh that is less uniform than that of the sphere.

III. Discussion

Both for the sphere and Thunderbird geometry, we observe a breakdown in convergence behavior at approximately 100 iterations after which the TFQMR method can apparently no longer decrease the relative residual norm (see inset in Fig. 1 and 4). This may be caused by a large condition number of the system matrix, even though the CFIE was used. Alternatively, a possible cause is the numerical roundoff error arising from the summation of billions of floating point numbers (as happens in the TFQMR algorithm). Unless special summation techniques such as Kahan summation are used, the relative errors caused by this are of the order of $\sqrt{N}\epsilon$ (with ϵ the machine precision) which happens to be around 0.006 in this case. Therefore, singular values of the system matrix below this value could be considered contaminated with roundoff error. In principle, this tentative cause for the convergence problem could be tested by adopting double precision calculations, at the cost of requiring twice as much memory. This is the subject for future research.

We can compare our results with [16], in which an MLFMA simulation with more than one billion unknowns is presented. In that work, a combined shared/distributed-memory approach was adopted. The workload was distributed according to the hybrid partitioning scheme and communication between nodes was handled using MPI. Further parallelization within each node was achieved using OpenMP. Because of this, fewer parallel processes (distributed-memory) are needed, leading to better ratios of iteration time and memory usage per unknown that were obtained in [16]. On the other hand, in contrast to the hybrid scheme, B-HiP is a weakly scalable algorithm, which is the crucial success factor for the simulation presented in this Communication.

IV. Conclusion

This Communication presents an implementation of the distributed-memory parallel MLFMA for electromagnetic scattering problems. By means of an improved hierarchical partitioning scheme, in which the radiation pattern sampling points are decomposed in both angular directions, a weakly scalable parallel algorithm is obtained. The weak scalability property is required to perform extremely large simulations. This Communication shows the solution, using $4\,096$ processes of two problems that consist of over respectively 3 and 2.5 billion of unknowns, the largest problems solved to date using high-frequency MLFMA. Additionally, it is demonstrated that the implementation achieves high parallel efficiencies.

Acknowledgment

The computational resources and services used in this work were provided by the Hercules Foundation and the Flemish Government – department EWI. Special thanks goes to Kenneth Hoste and Wouter Depypere for their kind support.

The work of B. Michiels was supported by a doctoral grant from the Special Research Fund (BOF) at Ghent University. The work of I. Bogaert was supported by a post-doctoral grant from the Research Foundation-Flanders (FWO-Vlaanderen).

References

[1] W.C. Chew, J. Jin, E. Michielssen and J. Song, *Fast and Efficient Algorithms in Computational Electromagnetics*. Boston, MA, USA: Artech House, 2001.

[2] S. Velamparambil, J.M. Song, W.C. Chew and K. Gallivan, "ScaleME: a portable scaleable multipole engine for electromagnetic and acoustic integral equation solvers," in *Proc. IEEE Antennas and Propagation Soc. Int. Symp.*, 1998, vol. 3, pp. 1774–1777.

[3] P. Havé, "A parallel implementation of the fast multipole method for Maxwell's equations," *Int. J. Numer. Meth. Fluids*, vol. 43, no. 8, pp. 839–864, Nov. 2003.

[4] F. Wu, Y. Zhang, Z.Z. Oo and E. Li, "Parallel Multilevel Fast Multipole Method for Solving Large-Scale Problems," *IEEE Antennas Propag. Mag.*, vol. 47, no. 4, pp. 110–118, Aug. 2005.

[5] J. Fostier and F. Olyslager, "An Asynchronous Parallel MLFMA for Scattering at Multiple Dielectric Objects," *IEEE Trans. Antennas Propag.*, vol. 56, no. 8, pp. 2346–2355, Aug. 2008.

[6] S. Velamparambil and W.C. Chew, "10 Million Unknowns: Is it that big?," *IEEE Antennas Propag. Mag.*, vol. 45, no. 2, pp. 43–58, Feb. 2003.

[7] S. Velamparambil and W.C. Chew, "Analysis and Performance of a Distributed Memory Multilevel Fast Multipole Algorithm", *IEEE Trans. Antennas Propag.*, vol. 53, no. 8, pp. 2719–2727, Aug. 2005.

[8] Ö. Ergül and L. Gürel, "Hierarchical Parallelisation Strategy for Multilevel Fast Multipole Algorithm in Computational Electromagnetics," *Electron. Lett.*, vol. 44, no. 6, pp. 3–4, 2008.

[9] Ö. Ergül and L. Gürel, "A Hierarchical Partitioning Strategy for an Efficient Parallelization of the MultiLevel Fast Multipole Algorithm,", *IEEE Trans. Antennas Propag.*, vol. 57, no. 6, pp. 1740–1750, Jun. 2009.

[10] L. Gürel and Ö. Ergül, "Hierarchical Parallelization of the Multilevel Fast Multipole Algorithm (MLFMA)", proc. of the IEEE, vol. 101, no. 2, pp. 332–341, 2013.

[11] C. Waltz, K. Sertel, M.A. Carr, B.C. Usner and J.L. Volakis, "Massively Parallel Fast Multipole Method Solutions of Large Electromagnetic Scattering Problems", IEEE Transactions on Antennas and Propagation, vol. 55, no. 6, pp. 1810–1816, 2007.

[12] J.M. Taboada, L. Landesa, F. Obelleiro, J.L. Rodriguez, J.M. Bertolo, M.G. Araujo, J.C. Mouriño and A. Gomez, "High Scalability FMM-FFT Electromagnetic Solver for Supercomputer Systems", IEEE Antennas and Propagation Magazine, vol. 51, no. 6, pp. 20–28, 2009.

[13] V. Melapudi, B. Shanker, S. Seal and S. Aluru, "A Scalable Parallel Wideband MLFMA for Efficient Electromagnetic Simulations on Large Scale Clusters," *IEEE Trans. Antennas Propag.*, vol. 59, no. 7, pp. 2565–2577, Jul. 2011.

[14] B. Michiels, J. Fostier, I. Bogaert and D. De Zutter, "Weak Scalability Analysis of the Distributed-Memory Parallel MLFMA," *IEEE Trans. Antennas Propag.*, vol. 61, no. 11, pp. 5567–5574, Nov. 2013.

[15] J. Fostier and F. Olyslager, "Provably Scalable Parallel Multilevel Fast Multipole Algorithm," *Electron. Lett.*, vol. 44, no. 19, pp. 1111–1112, Sept. 2008.

[16] X.M. Pan, W.C. Pi, M.L. Yang, Z. Peng and X.Q. Sheng, "Solving Problems with over One Billion Unknowns by the MLFMA", *IEEE Trans. Antennas Propag.*, vol. 60, no. 5, pp. 2571–2574, May 2012.

[17] F. Wei and A. Yilmaz, "A More Scalable and Efficient Parallelization of the Adaptive Integral Method – Part II: BIOEM Application,", *IEEE Trans. Antennas Propag.*, vol. 62, no. 2, pp. 727–738, Feb. 2014.

[18] G. Amdahl, "Validity of single-processor approach to achieving large-scale computing capability", in *Proc. AFIPS Conf.*, pp. 483–485, 1967.

[19] G. Mie, "Beiträge zur Optik trüber Medien, speziell kolloidaler Metallösungen," *Annalen der Physik*, vol. 25, no. 3, pp. 377–445, 1908.

[20] J.R. Mautz and R.F. Harrington, "H-Field, E-Field, and Combined-Field Solutions for Conducting Bodies of Revolution", *Archiv für Elektronik und Übertragungstechnik*, vol. 32, no. 4, pp. 157–164, Apr. 1978.