PROVISON OF REAL-TIME TROPOSPHERE DELAY CORRECTIONS BY UTILIZATION OF GENERAL PURPOSE GRAPHICS PROCESSING UNITS

Thomas HOBIGER[*], Ryuichi ICHIKAWA, Yasuhiro KOYAMA and Tetsuro KONDO[†]

National Institute of Information and Communications Technology (NICT)
Space-Time Standards Group
4-2-1 Nukui-Kitamachi, Koganei, Tokyo 184-8795, Japan

1. INTRODUCTION

Troposphere delays are one of the major error sources of space geodetic and remote sensing techniques. Unlike ionospheric delays, which have a dispersive characteristics and thus can be canceled out by dual-frequency measurements, atmospheric propagation effects have to be modeled within the post-processing or model adjustment stage. Many space-geodetic techniques apply so-called mapping functions which relate slant troposphere delays to zenith equivalents, that are estimated together with the target parameters. On the other hand, atmosphere delays within remote sensing techniques are neglected or considered only by simple models, which are not capable to represent larger areas or complex weather phenomena.

2. TROPOSPHERE SLANT DELAY CORRECTIONS FROM NUMERICAL WEATHER MODELS

Since numerical weather models have undergone tremendous improvements concerning accuracy and spatial resolution it has become feasible to utilize this information not only for the computation of mapping function coefficients, but also introduce ray-traced troposphere delays directly as corrections within the processing chain. [1] have shown that this strategy is capable to improve the performance of GNSS positioning, under the condition that small residual troposphere corrections are estimated within the adjustment process. Moreover, sophisticated and accurate ray-tracing algorithms have been developed [2], which properly geo-reference meteorologic data-sets and deliver troposphere slant delays by solving for the physical ray-propagation characteristics. In order to reduce computation time, simplified algorithms have been developed, which yield similar results, differing by less than a millimeter for usual observing geometries. Nevertheless, even fast these algorithms are not capable to meet the requirements of real-time applications, nor can they compute millions of rays, as needed for remote sensing applications, in reasonable time.

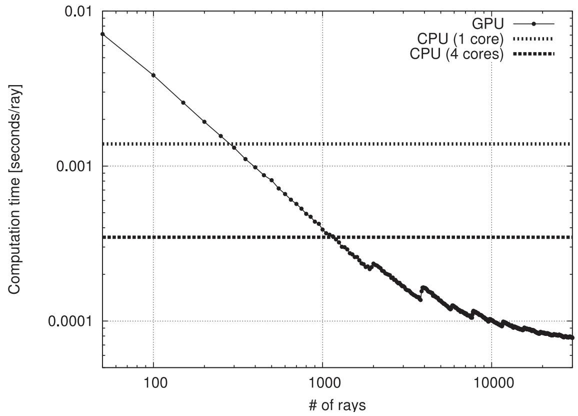
3. RAY-TRACING ON GENERAL PURPOSE GRAPHICS PROCESSING UNITS

Modern graphics processing units (GPUs) have not only become powerful parallel computing platforms, but are also supported by a variety of programming tools, which allow users to port their programs easily on the graphics processor [3]. Since ray-tracing can be done in parallel, the obvious step of carrying out the computation of troposphere delays on a GPU has been investigated. Given that modern graphic chips support full double precision representation one could expect nearly identical results as obtained on the CPU. One of the major drawbacks in the past was that for efficient computation the whole 4D weather model had to be uploaded into memory, which could not be achieved with the older graphic boards. Although the modern cards are sold with on-board memory of up to 1GB, the complete 4D refractivity field, obtained from the JMA meso-scale analysis model [4] still does not fit on GPU memory. Therefore, a new algorithm has been developed, which uploads only a sub-block to the GPU, ensuring that all rays are contained within this domain.

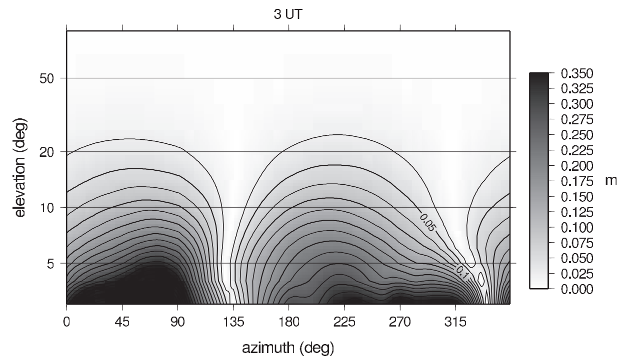
[*] Presenting author
[†] Currently on leave at Ajou University, Korea.

4. PERFORMANCE COMPARED TO COMPUTATIONS ON THE CPU

After simulating a random observation geometry for a ground GNSS receiver, the performance of the ray-tracing algorithms on the GPU respectively CPU can be compared with each other. Figure 1a shows the average computing time per ray, when being carried out either on the GPU, single-core CPU or a quad-core CPU using all four cores. As long as the number of rays is small, ray-tracing on the CPU is faster, but already for about 1000 rays the GPU starts to perform much better than the fastest CPU algorithms. If the number of rays exceeds 10,000, the GPU solution is in favour of the CPU, yielding identical troposphere slant delays on either device.



(a) Ray-tracing performance numbers.



(b) Absolute values of asymmetric troposphere slant delays.

Fig. 1. Figure (a) shows the performance of GPU and CPU ray-tracing computations measured in dependency on the number of total rays using a simulated data-set for GNSS station. Figure (b) depicts absolute asymmetric troposphere slant delays at Tsukuba, Japan on DOY 250, 2007. Ray-traced results reveal differences in the centimeter range for elevations up to 25 degrees.

5. FIELDS OF APPLICATION AND OUTLOOK

All space geodetic and remote sensing techniques which are utilizing signals within the microwave frequency band encounter atmosphere delays, which have to modeled within post-processing. Thus ray-traced troposphere delays offer an alternative to common processing strategies, given that the underlying numerical weather models are accurate enough to fulfill the requirements of the concerned technique. Moreover complex weather situations as for example depicted in Figure 1b, can be considered properly, and target parameters can be de-biased from influences caused by severe weather conditions. Ray-tracing on the GPU seems to be the way to go in the future, providing both, accurate troposphere slant delays and high computational performance, which delivers corrections to the user in near real-time and reduces processing time for a large number of rays, as e.g. necessary for InSAR.

6. REFERENCES

[1] T. Hobiger, R. Ichikawa, T. Takasu, Y. Koyama, and T. Kondo, "Ray-traced troposphere slant delays for precise point positioning," *Earth, Planets and Space*, vol. 60, no. 5, pp. e1–e4, 2008.

[2] T. Hobiger, R. Ichikawa, Y. Koyama, and T. Kondo, "Fast and accurate ray-tracing algorithms for real-time space geodetic applications using numerical weather models," *Journal of Geophysical Research*, vol. 113, no. D203027, pp. 1–14, 2008.

[3] T. Nguyen, *GPU Gems 3: Programming Techniques for High-Performance Graphics and General-Purpose Computation*, 2007.

[4] K. Saito, T. Fujita, Y. Yamada, J.I. Ishida, Y. Kumagai, K. Aranami, S. Ohmori, R. Nagasawa, S. Kumagai, C. Muroi, T. Kato, H. Eito, and Y. Yamazaki, "The Operational JMA Nonhydrostatic Mesoscale Model," *Mon. Wea. Rev.*, vol. 134, pp. 1266–1298, 2006.