Nearest Neighbor Searches on the GPU
A Massively Parallel Approach for Dynamic Point Clouds

Pedro Leite · João Marcelo Teixeira · Thiago Farias · Bernardo Reis · Veronica Teichrieb · Judith Kelner

Received: 6 February 2011 / Accepted: 20 August 2011 / Published online: 12 October 2011
© Springer Science+Business Media, LLC 2011

Abstract  We introduce a GPU grid-based data structure for massively parallel nearest neighbor searches for dynamic point clouds. The implementation provides real-time performance and it is executed on GPU, both grid construction and nearest neighbors (approximate or exact) searches. This minimizes the memory transfer between device and system memories, improving overall performance. The proposed algorithm may be used across different applications with static and dynamic scenarios. Moreover, our data structure supports three-dimensional point clouds and given its dynamic nature, the user can change the data structure’s parameters at runtime. The same applies to the number of neighbors to be found. Performance comparisons were made against previous works, endorsing the benefits of our solution. Finally, we were able to develop a real-time Point-Based Rendering application for validation of the data structure. Its drawbacks and data distribution’s impact on performance were analysed and some directions for further investigation are given.

P. Leite · J. M. Teixeira · T. Farias (✉) · B. Reis · V. Teichrieb · J. Kelner
Computer Science Center, Federal University of Pernambuco, Av. Professor Moraes Rego S/N, DINE,
1st floor, Cidade Universitaria, Recife, Pernambuco, CEP 50670-901, Brazil
e-mail: tsmcf@cin.ufpe.br

P. Leite
e-mail: pjsl@cin.ufpe.br

J. M. Teixeira
e-mail: jmxnt@cin.ufpe.br

B. Reis
e-mail: bfrs@cin.ufpe.br

V. Teichrieb
e-mail: vt@cin.ufpe.br

J. Kelner
e-mail: jk@cin.ufpe.br
Keywords  Nearest neighbor query · Massive parallel programming · KNN · ANN · Point-Based Rendering

1 Introduction

Spatial subdivision is a well-known technique for improving performance used in a variety of applications. There are many data structures that handle spatial subdivision efficiently. However, some data structures are well suited for specific problems. As an example, ray tracers need to do a lot of ray-triangle intersection tests in order to perform triangle culling. And kd-trees can provide a fast approach to solve such tests as shown in [1]. In addition, kd-trees can also be used for nearest neighbor search in photon mapping [2] or in point cloud modeling [3]. Other data structures such as octrees have been used for smoke and water simulation [4] and appearance preserving [5], while collision detection of deformable objects can be implemented with spatial hashes [6] or representative triangles [7]. Finally, different types of spatial subdivision data structures ease the task of culling non-visible objects from a scene [8].

Another common problem solved with an efficient spatial subdivision is the nearest neighbors, which consists on finding the closest neighbors to an input query. The neighbors and the input query can be described as a location, a car, a restaurant, etc. Nearest neighbor searches have their roots on the post-office problem, in which residences (input query) are assigned to their nearest post office (neighbor) and were first described by Donald Knuth [9]. Nowadays, a vast number of problems rely on nearest neighbor searches, including pattern classification [10], mobile information systems [11], implicit surfaces definition [12,13], simplification of point-sampled surfaces [14], nearest photon queries in photon mapping [2], nearest neighbor search in point cloud modeling and particle-based fluid simulation [3,15], normal estimation [16] and finite element modeling [17], among others. In this context, solving problems that rely on nearest neighbors searches implies in improving spatial subdivision techniques.

Nearest neighbor searches in dynamic point clouds comprise constructing a data structure (at each given timestamp) to hold the input data set as well as realize a lot of sorting, when searching for neighbors of a given set of query points. If the data structure does not intelligently subdivide the input data set in order to minimize such sorting, searches become slow and real-time criteria (a maximum processing time of 33 ms, including data structure construction) cannot be met. In other words, the main concern of this work is to address real-time massively parallel nearest neighbor searches in dynamic point clouds. This can be achieved through a grid data structure for both KNN (K nearest neighbors) and ANN (approximate nearest neighbors) searches and by exploiting the inherent parallel processing power of modern GPUs. In addition, a Point-Based Rendering (PBR) application is proposed, so the data structure can be tested and evaluated.

The remainder of this paper is organized as follows. In the next section we discuss some related work regarding KNN and ANN searches on CPU and GPU. Section 3 introduces the proposed solution to massively parallel nearest neighbor searches in dynamic point clouds. Both CPU and GPU implementations are highlighted. The obtained results are compared with each other as well as with previous works. Finally,