

# A Tiny World: Atom

CHUNG AN, CHEN, YANG, LIU, and LINXI, TAO

## ACM Reference Format:

Chung An, Chen, Yang, Liu, and Linxi, Tao. 2022. A Tiny World: Atom. 1, 1 (January 2022), 1 page. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

## 1 INTRODUCTION

Prior to the electron cloud model, there are a lot of attempts on building a theoretical model for an atom. Most of them, for example, the commonly known Rutherford-Bohr model, perceive the electrons as if they are planets orbiting the sun — a non-changing centripetal movement. The electron cloud, however, models an atom consisting of a small, yet massive when put aside to its electrons, nucleus surrounded by a non-deterministic cloud of electrons.

In the electron cloud model, electrons can theoretically be found anywhere in the space. However, they can be found more often in some regions, which we usually call them orbitals despite no orbital motions of electrons are being held, than others. A Carbon atom has two orbitals, where the inner layer contains two electrons and the outer layer contains four, and electrons are less likely to be found in between the two layers.

This project aims to simulate and render a Carbon atom's presence in a three-dimensional space. The simulation will be grounded in the more accurate electron cloud model, shown in Figure 1 below. An electron cloud model depicts the occurrence of an atom's electrons by a density map containing numerous sparse dots. In any region, the density of the dots draws a directly proportional relationship to the probability of an electron being present. To illustrate this model, We leverage a recently-built parallel programming language, namely **Taichi**, as our main development framework.

## 2 SOFTWARE

**Taichi** is a Python-based Domain-Specific Language designed for differentiable programming and speed up computer graphics development without compromising too much rendering performance. Despite inheriting most of its syntax from Python, **Taichi** relies on mostly parallel programming and does not carry over the downsides, for example, the slow computation speed, from Python.

**Taichi** is a higher level language such that it can be made to run on multiple mainstream rendering backends such as OpenGL, CUDA, and Metal. Using Pythonic decorators such as `@ti.kernel` and `@ti.func` brings the succeeding block of code into **Taichi**'s scope, where functions are naturally parallelized and differentiable on CPU or GPU devices. Two main data types in **Taichi** are Primitive Types

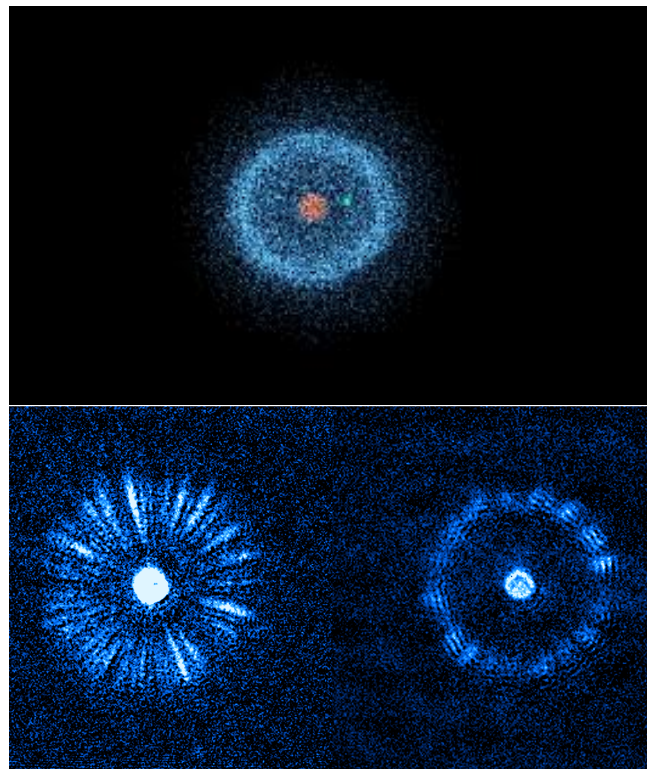


Fig. 1. Atoms and Their Nucleus and Electron Cloud

and Compound Types. Primitive Types are numerical data types used by backends, while Compound Types are user-defined types composed of multiple members. To bridge across **Taichi**'s scope and Python's Scope, we use a global variable object called `Field`. They can correspond to OpenGL's vectors or matrices depending on how users define them.

## 3 CODE AND DEVELOPMENT

We decided to proceed with render a Carbon atom to represent the main accomplishment of our work. However, we can easily scale up or down by editing the code to render other atoms as they only differ by the number of protons, neutrons, and electrons.

Authors' address: Chung An, Chen; Yang, Liu; Linxi, Tao.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

© 2022 Association for Computing Machinery.

XXXX-XXXX/2022/1-ART \$15.00

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>