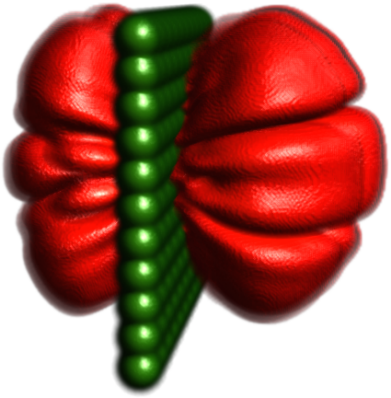
Time evolution simulation of the quantum mechanical wave function in 3D space

Zoltán Simon

Budapest University of Technology and Economics
Department of Control Engineering and Information Technology
Computer Engineering, M.Sc.
2. semester

Advisors: Dr. Balázs Csébfalvi[*], Dr. Péter Vancsó[†], Dr. Géza István Márk[‡]

November 2, 2023



[*]Budapest University of Technology and Economics, Department of Control Engineering and Information Technology
[†]Institute for Technical Physics and Materials Science, Centre for Energy Research, 1121 Budapest, Hungary
[‡]Institute for Technical Physics and Materials Science, Centre for Energy Research, 1121 Budapest, Hungary

Contents

Abstract

In quantum mechanics, the wave function describes the state of a physical system. In the non-relativistic case, the time evolution of the wave function is described by the time-dependent Schrödinger equation. In 1982, D Kosloff and R Kosloff proposed a method [1] to solve the time-dependent Schrödinger equation efficiently using Fourier transformation. The computational physics research group, led by Géza I. Márk in the Nanotechnology Departement, Institute for Technical Physics and Materials Science, Centre for Energy Research, in collaboration with Belgian researchers developed a simulation method based on three-dimensional wave packet dynamics for the study of the electron dynamics in nanosystems. A simplified, interactive two-dimensional version for educational purposes was published in 2020 [2]. In this work, we demonstrate two improvements of the wave packed dynamical simulation software: (i) the use of Graphical Processing Unit (GPU), which results in a vast (up to 50x) increase in simulation speed, and (ii) the introduction of advanced visualization techniques [3, 4] which are important to correctly interpret huge 4D space-time wave function data sets obtained from the simulation. We used our implementation to simulate typical quantum phenomena using wave packet dynamics. First, we tried the method on analytically describable cases, such as the simulation of the double-slit experiment, and then we investigated the operation of flash memory. We used raytraced volumetric visualization to render the resulting probability density. In our work, we introduce the basics of wave packet dynamics in quantum mechanics. We describe the used method in detail and showcase our simulation results.
For further information and animations, please visite
https://zoltansimon.info/src/content/research/wavepacketsim.html

1 Introduction

In the first quarter of the 20th century Quantum Mechanics (QM) opened a whole new window to understand our universe. Tamás Geszti, in his book [5] writes: *"learning QM is part of the process of understanding the world, and the person who masters it, understands the world better"*. It is a standard view that while Albert Einstein's General Relativity (GR) provides a model that accurately describes the laws of nature governing large-scale phenomena [6], quantum theory excels at atomic scale[1]. Although humanity has not yet accepted a single theory that would be capable of modeling both the small and large, thus bridging the gap between QM and GR, there are many use-cases where technology heavily depends on both theories. QM can be used efficiently to model the behavior of atomic particles. It describes how electrons behave in the orbitals around the atomic core and explains chemical reactions. It can be used to model the structure of molecules. QM made it possible for humanity to make use of nuclear energy in power plants, thus creating a new and efficient source of power. Computers –as we know them today– wouldn't be possible without a deep understanding of various quantum phenomena. These systems use transistors [7], which can also be thought of as a product of QM.

In nanotechnology, it is crucial to make quantum mechanical calculations to predict –and, in many cases, explain– the behavior of different nanostructures. One exciting field of study is the science of single-layer materials [8]. These are also known as 2D materials. One such carbon structure is called graphene [9, 10, 11]. This single-layered structure conducts heat and electricity very efficiently, thus raising high hopes in many when it comes to possible use-cases. Nowadays, quantum information science is getting a larger and larger audience [12]. Quantum Computing promises a newer seen increase in computation capabilities due to the massively parallel nature of quantum systems leveraging quantum superposition. Quantum Communication, on the other hand, opens new possibilities for secure information exchange with efficient channel encoding. Although both of the last two fields mentioned await technological evolution in order to be deployable at scale due to a large number of technical challenges, they both show that QM has a lot of real-life applications, and the spectrum of these will only become more colorful as technology evolves.

Inspired by the previously enumerated large amount of various fields of application, we set the goal to study the behavior of quantum systems by computer simulation. Such simulations are beneficial for scientists. They use such methods to accurately model the interaction between particles and various potential fields. In order to accomplish this goal, we choose a method that uses the Fast Fourier Transform (FFT) to efficiently calculate the time development of the quantum mechanical wave function. In QM, the wave function describes the state of a physical system. In the non-relativistic case, the time evolution of the wave function is described by the time-dependent Schrödinger equation [13]. In 1982, D Kosloff and R Kosloff proposed a method [1] to solve the time-dependent Schrödinger equation efficiently using Fourier transformation. In 2020, Géza István Márk published a paper [2] describing a computer program for the interactive solution of the time-dependent and stationary two-dimensional (2D) Schrödinger equation. Some details

[1] Under extreme circumstances such as in black holes macroscopic quantum phenomena are also present

of quantum phenomena are only observable by calculating with all three spatial dimensions. We found it worth stepping out from the two-dimensional plane and investigating these phenomena in three dimensions. Géza István Márk and his colleagues have already used 3D calculations in their research work [14, 15]. The difference is that their implementation uses solely the Central Processing Unit (CPU) of a computer. For visualization of the resulting probability density so far, they used the isosurface method. Our contribution mainly lies in leveraging the parallelization potential of the modern Graphical Processing Unit (GPU), thus significantly boosting the calculation speed by approximately a factor of 50 on our test hardware. We also apply state-of-the-art volumetric visualization techniques to create pleasing and comprehensible visuals to analyze the probability density evolution in 3D space. We believe that by combining the knowledge available for computer visualization specialists and physicists we can arrive to something greater than the possibilities available if each of us would be working only in his or her own domain of expertise. We write our implementation with future extendibility in mind since we would like to continue our work and develop a capable simulation platform that can be deployed as a tool for state-of-the-art scientific research.

In Section 2 of this work, we start by discussing the theoretical background. Then, in Section 3, we proceed to describe the used Fourier method to simulate the time development of the wave function. In Section 4, we provide an overview of the implementation details of our program. After this, in Section 5, we showcase our simulation results. Here, we compare the result of simple, analytically describable simulation scenarios to the mathematical model. At the end of Section 6, we summarize our results in a short discussion.

2 Theoretical background

2.1 Beginnings of quantum mechanical wave dynamics

In this section, we would like to give a brief introduction to the theory of QM. In this summary of physics history, we rely mainly on Tamás Geszti's QM book [5]. For English literature, we would like to point the reader's attention towards the critically acclaimed book of Claude Cohen-Tannoudji et al. [16].

In 1924, Louis de Broglie recognized that an electron moving with momentum of $p = M_e v$ can also be thought of as the propagation of some kind of a matter wave with λ wavelength.

$$\lambda = \frac{h}{p} \tag{1}$$

where h is called the Planck constant and equals to $h = 6,6 \times 10^{-34}\ Js$. This also explains the discrete energy levels in Bohr's model. The allowed orbitals are those where the wave is a single-valued function and thus closes into itself after one full rotation of 2π radians. If we use polar coordinates, this means that the wave has the same complex amplitude for coordinates α and $\alpha + n2\pi$ where $n = 0, 1, 2, \ldots$. Here we can further modify equation 1 and arrive to

$$\vec{p} = \hbar\vec{k} \tag{2}$$

where $\vec{k}$ is the wave vector and $\hbar = \frac{h}{2\pi}$ is the reduced Planck constant. The amplitude of this vector equals $\|\vec{k}\| = \frac{2\pi}{\lambda}$, and its direction is perpendicular to the wavefront.

Experiments show that matter waves exhibit self-interfering behavior similar to electromagnetic waves[2]. For example, the double-slit experiment, where a wave propagates through a barrier with a pair of two narrow slits and consequently creates an interference pattern on a canvas placed after the barrier, can be performed not only with light but a single electron as well. When the experiment is performed with electrons, the canvas is replaced with a measurement device that is able to register the incoming electron. By performing this experiment once, the measuring device will only register the electron at a single location. The interference becomes noticeable only after repeating the measurement multiple times since the distribution of the measured electron impact locations converges to the interference pattern. This behavior is also referred to as wave–particle duality because the particle interferes with itself, but also, there is only one single impact at the measuring device for each iteration of the experiment. Another important experimental observation is that if we create the linear combination of multiple matter waves and then the combined wave evolves over time than the resulting wave is precisely the same as if the originally combined waves would have evolved independently and we would only have combined the resulting waves. This latter property was also true for electromagnetic waves, and its called the superposition principle. Please note that although we keep comparing electromagnetic waves with matter waves, the two are different phenomena. One significant difference is that interference of electromagnetic waves happen due to different strength of electric and magnetic fields. In QM the wave function has a complex value thus, it can interfere even when only the phase is different and the amplitude is the same. Our goal in emphasizing the similarities is only to reinforce a possible intuition in those perhaps more familiar with the world of electromagnetism.

2.2 Wave packet dynamics

Erwin Schrödinger introduced the concept of the Wave Packet (WP) to show that in limiting cases QM gives back the classical particle model where small bullets are moving around and colliding. This particle-like propagation of a WP of the wave is due to the superposition of waves with different frequencies forming peaks. These waves extinguish each other for the most part, but at some position and time coordinates, the amplitudes add up to larger amplitude. An example of this is demonstrated in figure 1.

Ψ_k denotes the complex amplitude of a matter wave with $k = \frac{2\pi}{\lambda}$ wave number. The formula of a plane wave propagating in positive x direction can be described in the form of equation 3.

$$\Psi_k(x,t) = e^{i(kx-\omega t)} \tag{3}$$

Let's calculate the superposition of waves with different wave numbers! We assume that the angular velocity $omega(k)$ depends on the k wave number and that the waves with different wave

[2] An electron always only interferes with itself while light waves coming from multiple light sources can also interfere with each other.

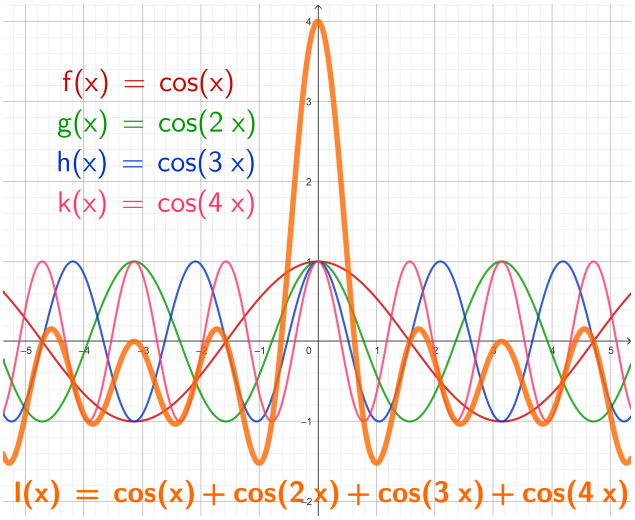


Figure 1: Superposition of cosine waves with different wavelength: the superposition forms a peak at the $x = 0$ coordinate

numbers have various $c(k)$ weights in the linear combination. This gives the following Fourier integral:

$$\Psi(x,t) = \int_k c(k) sin[kx - \omega(k)t]\ dk \tag{4}$$

By analyzing this formula, we can determine the propagation velocity of the amplitude peaks. In equation 4, the peaks form where and when the phase of the sine function is not dependent on the k integration variable. In other words, the derivative of the phase with respect to k is zero.

$$\frac{d}{dk}[kx - \omega(k)t] = x - \frac{d}{dk}\omega(k)t = 0 \tag{5}$$

By generalizing for all spatial coordinates and rearranging equation 5, we get the group velocity of the wave.

$$\vec{v}_{group} = \frac{\delta\omega(\vec{k})}{\delta\vec{k}} \tag{6}$$

By using the $E := \hbar\omega$ and $\hbar\vec{k} = \vec{p}$ substitutions for the energy and momentum, we arrive at the classically known formula for a particle of mass m

$$\vec{v}_{group} = \frac{\delta E(\vec{p},\vec{r})}{\delta\vec{p}} = \frac{\delta}{\delta\vec{p}}\left[\frac{\|\vec{p}\|^2}{2m} + V(\vec{r})\right] = \frac{\vec{p}}{m} \tag{7}$$

There are, however significant differences between the classical and QM model. The particle analogy comes with its limits. In 1927, Werner Heisenberg formulated his uncertainty principle. This states

that simultaneously, the position and the momentum of a particle can not be determined precisely. Only with a specific Δ precision. The result can be written in the form of equation 8.

$$\Delta x \cdot \Delta p \geq \frac{\hbar}{2} \tag{8}$$

A similar relation holds between time and energy.

$$\Delta t \cdot \Delta E \geq \frac{\hbar}{2} \tag{9}$$

2.3 The Schrödinger equation

So far, we know that atomic particles exhibit wave-like behavior, and in bounded systems, they can absorb or release energy only in discrete quanta. These matter waves can interfere with each other, and the superposition principle holds true. Along the way, we obtained a seemingly satisfactory energy, momentum, and velocity concept. One last step is to, instead of real amplitudes we allow complex amplitudes. This solves the problem of energy conservation.

All of these advancements led up to the formalization of the Schrödinger equation in 1926 by Erwin Schrödinger [13]. Already in the 19th century, in the field of optics, it was common practice to use linear partial differential equations to calculate light propagation. Linearity is a requirement for matter waves as well, since by the definition of superposition a general equation that aims to describe the behavior of matter waves adequately must be satisfied by the simple waves and also by the linear combination of these waves. Let us write the Ψ plane wave as a complex-valued function

$$\Psi(\vec{r}, t) = e^{i(\vec{k}\cdot\vec{r}-\omega t)} = e^{\frac{i}{\hbar}(\vec{p}\cdot\vec{r}-Et)} \tag{10}$$

First, let's assume that our wave is propagating in free space. The total energy of the particle consists only of kinetic part.

$$E = \frac{p^2}{2m} \tag{11}$$

We don't know the equation for the wave function yet, but it must incorporate this relation between the energy and momentum. In the wave function, both the energy and the momentum are in exponent. To express them, we have to take the derivative of the function with respect to position and time. The operation of calculating the partial derivative with respect to position is often referred to as calculating the gradient vector and is denoted by the ∇ ("nabla") symbol.

$$\begin{aligned} \frac{d}{dt}\Psi(\vec{r}, t) &= -\frac{i}{\hbar}E\ \Psi(\vec{r}, t) \\ \frac{\delta}{\delta\vec{r}}\Psi(\vec{r}, t) &= \nabla\Psi(\vec{r}, t) = \frac{i}{\hbar}\vec{p}\ \Psi(\vec{r}, t) \end{aligned} \tag{12}$$

In the kinetic energy equation 11, we actually need the square of the magnitude of momentum. To calculate it, we introduce the $\Delta = \vec{\nabla} \cdot \vec{\nabla}$ Laplace operator. By combining equation 11 and 12 we

can arrive to the following linear differential equation

$$i\hbar\frac{d}{dt}\Psi(\vec{r},t) = -\frac{\hbar^2}{2m}\Delta\Psi(\vec{r},t) \tag{13}$$

This is the Schrödinger equation for matter wave propagating in free space. Let's generalize this by eliminating the previous assumption that the total energy is solely kinetic. Now, we allow an $V(\vec{r})$ localized potential energy to be an arbitrary function of location. Thus, the total energy becomes

$$E = \frac{p^2}{2m} + V(\vec{r}) = \mathcal{H}(\vec{p},\vec{r}) \tag{14}$$

Here, we introduced the $\mathcal{H}$ notation to express that this is a form of the so-called Hamiltonian function named after William Rowan Hamilton, who used his formalism to describe classical mechanics in the 19th century [17]. Using equation 14 to modify equation 13, we trivially arrive at the more general form of the Schrödinger equation

$$\begin{aligned} i\hbar\frac{d}{dt}\Psi(\vec{r},t) &= -\frac{\hbar^2}{2m}\Delta\Psi(\vec{r},t) + V(\vec{r})\Psi(\vec{r},t) \\ \frac{d}{dt}\Psi(\vec{r},t) &= -\frac{i}{\hbar}\hat{H}\Psi(\vec{r},t) \end{aligned} \tag{15}$$

We can see that on the left side, we basically take the first derivative of the wave function with respect to the time and on the right side we let a $\hat{H} = -\frac{\hbar^2}{2m}\Delta + V(\vec{r})$ operator affect the wave function. By solving this differential equation and specifying an initial state, we can predict the time development of a quantum mechanical wave function.

The only remaining question also meaningful for our simulation is that how exactly should we interpret the resulting function? What's the actual physical meaning behind the amplitude of the wave? Experiments show that the square of the absolute value of the complex amplitude is the probability density associated with the particle being found in a given infinitesimally small portion of space at a given time. For convenience and to be sound with probability theory, we normalize the amplitude of the wave function so that the velocity of the particle being found "somewhere" in space equals $\mathbb{P} = 1$.

$$\int_{\mathcal{V}} |\Psi(\vec{r},t)|^2 \, d^3r = 1 \tag{16}$$

3 Calculation method

3.1 Survey and comparison of available methods

In our simulation, we numerically solve the time-dependent Schrödinger equation. There are multiple available algorithms for this task. A widely used category of solving strategies are the Finite Difference in Time Domain (FDTD) methods. This approach is also widely adopted to simulate the solution of Maxwell equations [18, 19]. FDTD technique for the analysis of quantum devices

(FDTD-Q) however exhibit numerical instability [20]. After a longer sequence of simulation steps, the solutions diverge. There are various alterations to the basic method. One such method was proposed by Min Zhu et al. [21] in the year 2014. Their algorithm is called the Runge-Kutta High-Order Finite Difference in Time Domain (RK-HO-FDTD). In 2021, a follow-up paper [22] demonstrated the superiority of RK-HO-FDTD over the basic FDTD-Q approach. In the same year, Frederick Ira Moxley et al. [23] proposed another method based on FDTD-Q. These methods solve the majority of problems related to FDTD-Q.

However, in our work, we use a different approach. Back in 1982, D Kosloff and R Kosloff proposed a method [1] to solve the time-dependent Schrödinger equation efficiently using Fourier transformation. The advantage of this algorithm is the high numerical stability of the time evolution step. In the adopted FFT method, no signs of divergence are present even after a large number of simulation steps. The time development step of the algorithm has a time complexity of $\mathcal{O}(N \log N)$ since it only uses six FFT runs ($\mathcal{O}(N \log N)$ each) and three element-wise multiplication between tensors ($\mathcal{O}(N)$ each). The amount of FFT runs and multiplications can be reduced further if we do not want to read the results of the time development in each step. A significant speed-up can be reached by using a parallelized implementation of the FFT algorithm as we did by using an efficient GPU implementation.

3.2 Detailed explanation of the split-time Fourier method

In the following part, we would like to explain the FFT method in detail. Let us start by deriving the formal solution of the 15 differential equation.

$$\begin{aligned} \int_{t_0}^{t} \frac{d}{d\tau}\Psi(\vec{r},\tau)\, d\tau &= \int_{t_0}^{t} -\frac{i}{\hbar}\hat{H}\Psi(\vec{r},\tau)\, d\tau \\ \Psi(\vec{r},t) - \Psi(\vec{r},t_0) &= \int_{t_0}^{t} -\frac{i}{\hbar}\hat{H}\Psi(\vec{r},\tau)\, d\tau \\ &\vdots \\ \Psi(\vec{r},t) &= e^{-\frac{i}{\hbar}\hat{H}(t-t_0)}\Psi(\vec{r},t_0) \end{aligned} \tag{17}$$

where $\Psi(\vec{r},t_0)$ is a specified initial state and $\Psi(\vec{r},t)$ is the state after some $\delta t = t - t_0$ time. The problematic part is the Hamiltonian operator in the exponent. The kinetic and potential operators can not be commuted in general. Hence, exponential can not be factored. We can decomposes the exponential by the symmetrical unitary product [24, 25] as shown in the form 18.

$$e^{-\frac{i}{\hbar}\hat{H}\delta t} = e^{-\frac{i}{\hbar}(\hat{K}+\hat{V})\delta t} \approx e^{-\frac{i}{\hbar}\hat{K}\delta t/2}\, e^{-\frac{i}{\hbar}\hat{V}\delta t}\, e^{-\frac{i}{\hbar}\hat{K}\delta t/2} \tag{18}$$

The error of this approximation is $\mathcal{O}(\delta t^3)$; therefore, we have to be careful with selecting a small enough time resolution. When the potential energy is localized, the $\hat{V}$ operator is a simple multiplication with $V(\vec{r})$ function; thus the middle part of the product is as follows

$$e^{-\frac{i}{\hbar}\hat{V}\delta t}\Psi = e^{-\frac{i}{\hbar}V(\vec{r})\delta t}\Psi \tag{19}$$

The $\hat{K}$ kinetic operator involves calculating the spatial derivative of the wave function.

Previously in equation 12 we have seen that calculating the spatial gradient $\nabla\Psi(\vec{r},t)$ yields multiplication by $\frac{i}{\hbar}\vec{p} = i\vec{k}$. We only need to get the value of the wave vector. This is when the Fourier transform comes into play, which transforms between real space and momentum space. The following relation holds for the derivative of an arbitrary f function and its Fourier transform

$$ik\mathcal{F}\{f\} = \mathcal{F}\{f'\} \tag{20}$$

Taking the derivative in real space means multiplication by ik imaginary wave number in momentum space. We work with the $\Delta = \nabla \cdot \nabla$ Laplace operator, so we have to multiply by $(ik)^2 = -k^2$. By exploiting the linearity of the Fourier transform, we arrive at the following formula for the kinetic energy part of the Hamiltonian function

$$\hat{K}\Psi = \frac{p^2}{2m}\Psi = -\frac{\hbar^2}{2m}\Delta\Psi = -\frac{\hbar^2}{2m}\mathcal{F}^{-1}\{-k^2\mathcal{F}\{\Psi\}\} \tag{21}$$

where $\mathcal{F}^{-1}$ is the inverse Fourier transform. In momentum space, the k wave number is trivially given as it can be thought of as the very coordinate the functions are parameterized with. Having that said, it is essential to distinguish between ν frequency, λ wavelength, k wave number, and even ω angular velocity. These relate as follows

$$k = \frac{2\pi}{\lambda} = \frac{2\pi\nu}{v_{group}} = \frac{\omega}{v_{group}} \tag{22}$$

where v_{group} denotes the group velocity of the wave packet.

Actually, in equation 18, the $\hat{K}$ kinetic energy operator is in the exponent multiplied by $-\frac{i}{\hbar}\delta t/2$. Using the knowledge gathered from equation 21, we can now write

$$e^{-\frac{i}{\hbar}\hat{K}\delta t/2}\Psi = \mathcal{F}^{-1}\left[e^{-\frac{i}{\hbar}\left(-\frac{\hbar^2}{2m}\right)(-k^2)\delta t/2}\mathcal{F}\left[\Psi\right]\right] = \mathcal{F}^{-1}\left[e^{-\frac{ik^2\hbar\delta t}{4m}}\mathcal{F}\left[\Psi\right]\right] \tag{23}$$

Before writing the pseudo-code, let us write the wave function as a function of discrete $i_j \in \{0, 1, \ldots, N_j - 1\}$ coordinates where $j \in \{x, y, z, t\}$ and N_j are the number of discrete grid points along each axis.

Now $\vec{r}_x = i_x\delta x$, $\vec{r}_y = i_y\delta y$, $\vec{r}_z = i_z\delta z$ and $t = i_t\delta t$ are the original spatial and time coordinates and δj is the resolution of the different dimensions respectively. With the new discretized coordinates, the Schrödinger equation 15 can be written in the following form

$$i\hbar\frac{d}{dt}\Psi(i_x, i_y, i_z, i_t) = -\frac{\hbar^2}{2m}\Delta\Psi(i_x, i_y, i_z, i_t) + V(\vec{r})\Psi(i_x, i_y, i_z, i_t) \tag{24}$$

Having a discrete data set, Discrete Fourier transform (DFT) can be efficiently implemented using the Fast Fourier Transform (FFT) algorithm. The output of the simulation is the wave function. The probability density can be obtained by calculating the square of the absolute value of the wave

function for each grid cell which is the same as calculating the standard scalar product between $\langle\Psi|$ and $|\Psi\rangle$ vectors where $\langle\Psi|$ is the transposed complex conjugate of $|\Psi\rangle$ as seen in equation 25.

$$p(\vec{r},t) = |\Psi(\vec{r},t)|^2 = \langle\Psi(\vec{r},t)|\Psi(\vec{r},t)\rangle \tag{25}$$

Making use of formulas 18, 19 and 23 and plugging them into the formal solution of the Schrödinger equation we can create an algorithm for the time development of the wave function. We will use the Ψ to represent a complex-valued tensor where the elements of this tensor can be obtained by using the i_j discrete indices. The algorithm can be written in the form of the following pseudo code

Algorithm 1 Time advance algorithm

$\Psi \leftarrow$ initial state of the wave function
$V \leftarrow$ localized potential
$\delta t \leftarrow$ time resolution
$N_t \leftarrow$ number of time steps
for $i \in [0, N_t)$ **do**
  $\Psi^{(1)} \leftarrow FFT^{-1}\left[P_K \; FFT\left[\Psi\right]\right]$
  $\Psi^{(2)} \leftarrow P_V \; \Psi^{(1)}$
  $\Psi \leftarrow FFT^{-1}\left[P_K \; FFT\left[\Psi^{(2)}\right]\right]$
  Visualize $|\Psi|^2$
end for

Here $P_K(i_x, i_y, i_z) \sim e^{-\frac{i\hbar\delta t}{4m}\left[(2\pi i_x/N_x/\delta x)^2+(2\pi i_y/N_y/\delta y)^2+(2\pi i_z/N_z/\delta z)^2\right]}$ is the kinetic energy propagator in momentum space and $P_V(i_x, i_y, i_z) = e^{-\frac{i}{\hbar}V(i_x,i_y,i_z)\delta t}$ is the potential energy propagator in real space. We used the discretized coordinates to express the same operations as in 19 and 23. Please note that in the kinetic propagator's case, it is important to follow the convention used by the specific FFT implementation. The formula written here is not entirely correct. The FFT used in our implementation puts the amplitude associated with the largest representable frequency of $\frac{1}{2N}$ in the middle of the tensor. In the second half of the tensor for each axis, the amplitudes are for the negative frequencies in ascending absolute value so that the largest index represents the $-\frac{1}{N}$ frequency. This means that we had to modify the definition of P_K accordingly. One way to make the correction is to check whether $i_j/N_j > \frac{1}{2}$ is true. If it is, then modify it like $i_j := N - i_j$.

3.3 Defining Gaussian wave packets

In the algorithm, first, we have to specify an initial state for the wave function. Erwin Schrödinger introduced the concept of the Wave Packet (WP). A WP is a wavefront that propagates and reflects as a classical particle would do and also exhibits all the wave-like behavior described by QM. It bridges the gap between classical and quantum physics. The term Wave Packet Dynamics (WPD) refers to the process of modeling QM systems by initializing WPs and observing the propagation,

reflection scattering, and interference of the WP. In our work, we use Gaussian WPs. This probability density of this WP has Gaussian distribution [26], hence the name. The definition of such a packet can be written in the following form

$$\psi(\vec{r};\ a, \vec{r_0}, \vec{k_0}) = \left(\frac{2}{\pi a^2}\right)^{\frac{D}{4}} \cdot exp\left(i\vec{k_0} \cdot \vec{r}\right) \cdot exp\left(-\frac{|\vec{r} - \vec{r_0}|^2}{a^2}\right) \tag{26}$$

where $\vec{r_0}$ is the initial position (with the highest probability density), $\vec{k_0}$ is the initial wave vector, and D is the dimension, which is $D := 3$ in our simulation. We can obtain the width of the Gaussian WP as $\Delta r = \frac{a}{2}$.

3.4 A simple trick to optimize the algorithm

If we do not want to visualize the probability density in each iteration, we can further optimize the calculation by merging the first step of the nth iteration and the last step of the $(n-1)$th iteration. If we omit the visualization step, we can do one forward FFT then perform a multiplication between the moment space wave tensor and the P_K^2 kinetic propagator calculated for a whole δt interval – instead of the one used in Algorithm 1 calculated for $\delta t/2$ interval– then do the backward FFT. The pseudo-code of this optimized algorithm is written below

Algorithm 2 Optimized time advance algorithm without visualization

$\Psi \leftarrow$ initial state of the wave function
$V \leftarrow$ localized potential
$\delta t \leftarrow$ time resolution
$N_t \leftarrow$ number of time steps
for $i \in [0, N_t)$ **do**
  $\Psi^{(1)} \leftarrow FFT^{-1}\left[P_K^2\ FFT\left[\Psi\right]\right]$
  $\Psi^{(2)} \leftarrow P_V\ \Psi^{(1)}$
end for

3.5 Free space boundary condition

When working with DFT it is vital to take into account the boundary conditions of the transformed region. The DFT implies a periodic boundary condition in which value at $i_j = 0$ connects smoothly to values at $i_j = 0$ for each of the respective $j \in \{x, y, z\}$ dimensions. If the data transform does not satisfy this condition, unrealistic artifacts arise. One technique to battle this requirement is to define a high potential wall on all the edges of the simulated volume so that the wave packet bounces back and never reaches the problematic edge. This, however, limits the possible simulation cases since we can not simulate scenarios where the wave packet would leave the volume. The wave packet reflected by the boundary potential interferes with itself, thus potentially disturbing the observation of the object of simulation. In our implementation, we used another approach called

free space boundary condition. This fixes the issues arising from the periodic boundary condition and the potential box. We extend the simulated volume beyond the observed part and outside the visible box, and we create a draining potential. This draining potential gradually increases towards the edge of the simulated volume. As the name suggests, its effect on the wave packet is that it diminishes its amplitude. By initializing a high enough draining potential, the unrealistic artifacts on the edge of the simulated volume are practically negligible. In equation 19, we have seen that the potential propagator is a multiplication by a unit length complex number with phase proportional to $V(\vec{r})$ localized potential. This normally only introduces a rotation on the complex plane. We can, however incorporate the desired draining effect by introducing complex potential. Let us examine what happens if a complex number has a complex phase!

$$e^{i|z|e^{i\gamma}} = e^{i|z|[cos(\gamma)+isin(\gamma)]} = e^{i|z|cos(\gamma)-|z|sin(\gamma)} = e^{i|z|cos(\gamma)}e^{-|z|sin(\gamma)} \tag{27}$$

In equation 27, we showed that by defining an imaginary potential, we can reduce the magnitude of the wave function if we then apply the potential propagator. In our simulation, we initialized the draining potential to be zero in the observed region, and right from the corners of this box, it begins to increase as a quadratic function of distance from the center of the simulated volume. It reaches its maximum in the four corners of the simulated volume.

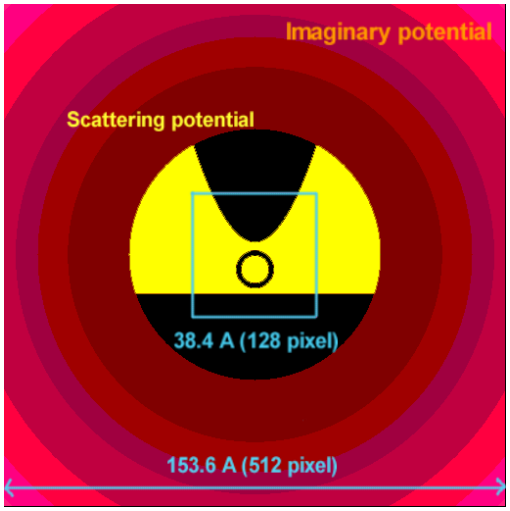


Figure 2: Calculation scheme used for the simulation of a Scanning Tunneling Microscope (STM) investigation of a carbon nanotube [hivatkozás az 1998-as PRB cikkre]. The upper black ellipse is the STM tip, the middle black ring is the cross-section of the nanotube, and the lower black region is the support electrode. The red concentric circles show the imaginary potential with increasing magnitude.

4 Our implementation

4.1 Short summary of the functionality of our simulator

Using the Fourier method, we created a Python application simulating the time development of the quantum mechanical wave function. We use ray tracing to visualize the resulting volumetric probability density. The visualization requires the sampling of a 3D data set on a discretized grid. This makes it impossible to fully reconstruct the wave function that we simulated using only a finite resolution to begin with. In order to fight sampling artifacts, we deploy a state-of-the-art triquadratic reconstruction filter recently proposed by Balázs Csébfalvi [4].

4.2 Technologies in use

We choose Python [27] as our programming language because there is a vast amount of helpful tools implemented to use with it that specifically aim towards leveraging the difficulty of writing mathematical and physics-related simulations. The majority of these tools are written in hardware-friendly languages such as Fortran, C, or C++. However, they come with an easy-to-use Application Programming Interface (API) that can be accessed from high-level programming languages. In these modern languages, such as Python, one works on a higher abstraction level, usually not dealing with memory management or pointer arithmetic. Out implementation heavily depends on the Numpy library [28]. NumPy is a library that can be utilized to work efficiently with large arrays and tensors performing computationally intensive operations. A similar library is SciPy [29], also used in our program. For 3D visualization, we choose to use VisPy [30]. This library provides an excellent basic set of features to handle virtual cameras and create scenes but also enables us to go deeper and write our own GPU shader code. We made use of this to modify the default volume visualization code to fit our needs.

4.3 GPU parallelization and Just In Time compilation

The Fourier method described in section 3 opens up the possibility to implement the simulation on the GPU. Using GPU acceleration is one of our contributions to the already existing implementation used by Géza István Márk and Péter Vancsó. The Compute Unified Device Architecture (CUDA) toolkit is often used for parallel computational tasks implemented on the GPU [31]. It comes with a powerful GPU based FFT implementation. To use CUDA with Python, we selected the CuPy wrapping library [32] that provides abstraction over CUDA. We have used Numba to access Just In Time compilation (JIT) features. JIT means that for some parts of the otherwise interpreted source code, the compiler performs a runtime translation to native code. This code then runs efficiently on the CPU. This feature is especially useful when iterating over large arrays. We utilize JIT when initializing the WP, localized potential and propagators. For other purposes, we use multiple other libraries such as Imageio, Matplotlib, toml, Pillow, Keyboard, Colorama, Tqdm, PyQt5, and PyOpenGL. The sources for these libraries can be found on the Internet. We have listed the required versions in a requirements.txt file in the project's GitHub repository.

4.4 About the program's user interface

In its current state, our application has only a console interface, and it saves the resulting images and videos into files. The reason that we have not prioritized the development of a graphical user interface is that we think that terminal interfaces can still have their benefits even in 2023. An application that only requires a terminal to function is more straightforward; thus, more effort can be made to improve the core functionality. The audience of this software is scientists and engineers in the first place. This is especially true in the early stage of development that we are currently in. It already has, however a snapshot system that makes it possible to interrupt a more extended simulation and later resume from the exact state where it was previously halted. This turned out to be a very convenient feature, although since we use GPU acceleration, the simulation times are generally much shorter. Our design philosophy dictated communicating as much information about the simulated wave function to the user as possible. This direction can be thought of as controversial since too much data can obfuscate the critical details, especially in a plain console print. We try to battle this effect by using colorful prints and by saving the text into a log file so that the parameters can be found even after the simulation has finished.

To specify the parameters of a simulation, we use a configuration file in Tom's Obvious Minimal Language (TOML) format [33]. The flexibility and universality of this data format made it an overall good choice. We can use a single TOML file to set the resolution and dimensions of the simulated volume, specify the parameters of the wave packet, and add an arbitrary number of various potential barriers. In this file, we have parameters to configure the 3D visualization and others.

4.5 Generated output

As we have mentioned earlier, the output of the program gets saved in files. The program creates images, videos, and text files. The images can be thought of as higher-resolution snapshots from the videos, but we also create images that do not correspond to any of the videos. Currently, we generate five types of different images. We call these *Canvas probability*, *Canvas dwell time*, *Per axis probability density*, *Probability density 3D*, and finally, *Probability evolution*. In each of these categories, a sequence of images is generated for each run of the simulation. The interval at which the state of the simulation is captured can be specified in the above-described configuration file. *Canvas probability* is the $\rho(\vec{r}, t)$ probability density in a specified plane intersecting the volume. $\tau(\vec{r})$ *canvas dwell time* is the integral of the $\rho(\vec{r}, t)$ probability density over t elapsed time in a specified plane intersecting the simulated volume. This can be formalized as

$$\tau(\vec{r}, t) = \int_0^t \rho(\vec{r}, \delta)\ d\delta \tag{28}$$

For now, the measurement plane used in the above-described methods has to be perpendicular to one of the three principal axes to simplify the calculation. Both *dwell time* and *probability density* can be used to visualize interference patterns after a wave packet passes through some kind of a

potential barrier such as a double-slit or a diffraction grating, as we will show in Section 5. *Per axis probability density* plots integrated probability densities for each axis. To obtain the plot of $\rho_X(x,t)$

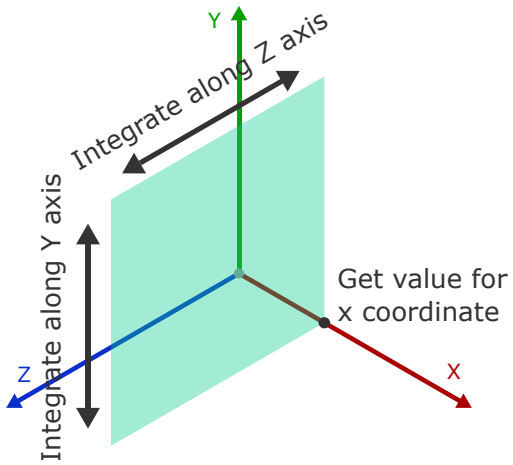


Figure 3: Calculation method of the *Per axis probability density* for a given x coordinate

per axis probability density, we integrate the $\rho(\vec{r},t)$ where $r_x = x$ and the other two coordinates run across their domain. Equation 29 gives the formula for calculating this measurable for the X axis.

$$\rho_X(x,t) = \int_Y \int_Z \rho(\vec{r},t)\ dydz \tag{29}$$

Figure 3 attempts to provide further intuition. We also overlap the plot of the potential barriers on the same graph to understand the changes in the propagation of the wave packet caused by the localized potential. This type of plot can provide helpful information for most simulation cases. It is beneficial when we want to simulate the behavior of three one-dimensional particles in a three-dimensional configuration space. In this case, the projected probability density along each axis represents the wave packet of a different 1D particle. It is possible to initialize a potential that models the collision between these three particles. This creates the effect that the particles interact. In the configuration file, there is a possibility to set whether a potential barrier should appear in the visualization. By disabling the visualization of the interaction potential, we can get rid of any hardly conceptualizable elements of the resulting potential and maybe focus on a wall or a harmonic oscillator instead. Probability density 3D is the most self-explanatory output of the system. Here, we take the probability density as a 3D volumetric data set and visualize it using ray tracing. Ray tracing is the method where we conceptually shoot rays of light from the virtual camera through the visualized volume. Figure 4 explains the concept of ray tracing. For each pixel of the image, there is exactly one ray. We progress along the ray in discrete steps, and at each step, we sample the data set. The data read from the volume usually has a single intensity value. We interpret the data using a transfer function that determines the color and opacity that should be associated with the intensity. The color and opacity are then combined along the ray, resulting in the final color and

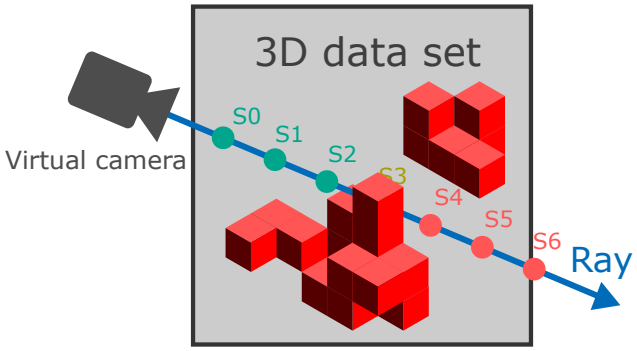


Figure 4: Visual explanation of the ray tracing method: we sample the 3D data set along rays shot from the virtual camera

opacity value for the image's pixel. The operator that recursively combines the color and opacity values is called under operator. For recursive calculation of the A_n opacity in nth step along the ray, it can be written in the form of

$$A_n = A_{n-1} + \alpha_n(1 - A_{n-1}) \tag{30}$$

where A_{n-1} is the opacity calculated at the previous step and α_n is the local opacity at the nth step. A similar equation could be written for red, green, and blue (RGB) color channels. If we would only use the colors obtained by simply stepping through the sample points, the look of the probability density would be rather dull. We also utilize approximated gradient vectors. One can calculate these vectors by taking the central difference using six additional samples around the currently sampled point. (We, however, deployed a better method described later.) The gradient vector points towards the direction in which the probability density increases the fastest. If we make the analogy that a solid body object also has gradient vectors right on the edge of its surface so that these vectors point towards the inside of the object perpendicular to the surface, we can make the connection between gradient vectors and surface normals. Normal vectors of a surface are unit-length vectors pointing towards the outside world from the surface and are also perpendicular to the surface. Normal vectors can be expressed as negated and normalized gradients. Normal vectors are often used in various surface shading models since they give a good description of how surfaces reflect light. Even though our probability density has a much less clearly defined gradient, this technique can still be used to obtain an approximated normal vector. If this vector is sufficiently defined (the magnitude of the gradient is not too close to zero), then we can use a simple shading model such as the Blinn-Phong reflection model [34] to enhance our visuals. We even use the length information of the gradient to fade between the shaded and unshaded look. We would like to mention that in our previous works, we have experimented with more physically plausible shading techniques, such as applying the Cook-Torrance Bidirectional Reflectance Distribution Function (BRDF) [35]. This, however, has not yielded good results for volumetric visualization since surface imperfections are calculated into the model; thus, the generally fussy nature of volumetric data

sets results in an overall dark look. The simple Blinn-Phong model, however, is cheap to compute and provides sufficiently strong specular highlights while it's still easy to parameterize. For most of our simulations, we use a total simulated volume of $512 \times 512 \times 512$ data points. We leave the outer half of the total volume to accommodate the draining potential described in the last part of section 3. This means that so far, we only visualize 50% of the stored data set: $256 \times 256 \times 256$. In the future, we may experiment with reducing the space used up by the draining potential. Until then, we rely heavily on good reconstruction filters to get rid of the aliasing effects visible by sampling the data set of relatively low resolution. We utilize Balázs Csébfalvi's reconstruction filter [4]. This method provides consistent gradient estimation, that is, the analytic gradient of the reconstructed function. It has an approximation order of three, and it is also C^1 continuous. This method evaluates eight trilinear samples to calculate the intensity and the gradient for a given point. In the current generation of GPUs trilinear filtering can be efficiently evaluated using the built-in sampling routines. The eight is only one more sample compared to the seven used in the less advanced intensity plus central difference technique, and we also get better results, as shown in the referred paper.

The last type of the five different image outputs of our program we call *Probability evolution*. To create this output, we divided the visible region of the simulated volume into separate partitions. On these images of this type, we display a plot where we integrate the probability density over these regions following equation 31.

$$\mathbb{P}(t) = \int_{\mathcal{V}} \rho(\vec{r}, t) d^3 r \tag{31}$$

Conceptually, this gives us the probability of the event of the particle being found in these spatial regions. It is also allowed to specify overlapping regions. By doing so, we only get the probability of non-excluding events. We prefer to create a measurement region for the entirety of the visible volume. This is very informative when the particle leaves the visible parts. Hence, the probability of the particle being found somewhere inside the visible part drops below one. Note that modeling such scenarios is made possible by using the draining potential.

Besides the sequence of images, our program also renders two animations in MP4 format. One is created from a sequence of images rendered as in the *Probability density 3D* sequence. The other is the animated version of the *Per axis probability density* sequence. The rate at which the state of the simulation is captured as an animation frame and the playback frame rate can be configured separately.

4.6 Performance test

We measured the performance of our application. We used a personal laptop to run and test the program. The system specification of our computer is summarized in figure 5. First, we tried a CPU-only version of our simulator to compare the results with the GPU accelerated implementation. The results of the comparison can be found in figure 6. Here, we tested three different configurations with varying resolutions. We measured the average iteration count per second. The test shows that by using GPU acceleration, we obtained significant speed up over the CPU-only implementation.

CPU	AMD Ryzen 5 6600H with Radeon Graphics 3.30 GHz
GPU	NVIDIA GeForce RTX 3050 Ti Laptop GPU
RAM	16 GB
OS	MS Windows 11 64-bit

Figure 5: System specification of the used test hardware

Input size	CPU-only Avg. [iter/s]	GPU accelerated Avg. [iter/s]	Avg. speed up
$128 \times 128 \times 128$	1.1	11.5	$\times 10.45$
$256 \times 256 \times 256$	0.09	6.5	$\times 72.22$
$512 \times 512 \times 512$	0.01	0.5	$\times 50.00$

Figure 6: Results of a performance test using a CPU-only and a GPU accelerated version of the application averaged

5 Results

5.1 General approach and the system of units

We used our software to perform various WPD simulations. In this section, we present the results of some of these simulations. We will shortly introduce the simulated scenarios. We also want to provide some formulas to validate the correctness of some of the simpler scenarios analytically. Our simulator uses the Hartree atomic units [36]. Every quantity in the upcoming part should be interpreted as such. This unit system makes it convenient to deal with quantities at atomic scale. You can find animations showing the time evolution of WPs simulated by our software on https://zoltansimon.info/src/content/research/wavepacketsim.html. From this web page, you can also navigate to the GitHub repository of our project, where all our source code is available for download.

5.2 Double-slit experiment

First, we would like to present the simulation results of an electron scattering experiment. Scattering of a particle happens when the WP of the particle passes through some kind of a barrier with holes in it. In our simulation, we can model the barrier as a localized potential. The WP arrives from one side of the barrier. While passing through this barrier, it scatters, and some of the WP gets reflected back. The portion of the WP that passed through –suffering scattering– continues forward and consequently arrives at a measuring device[3]. In our simulation, our measuring device is a virtual

[3]In scattering and diffraction experiments we can make the distinction between a near field and far field solution.

canvas where we measure the probability density. A simple scattering scenario is the double-slit experiment. Here the barrier is a potential wall with two narrow parallel slits. The WP passes through these slits. According to the Huygens–Fresnel principle in scattering experiments, we can approximate the wave function after the barrier as concentric waves emitted from the slits or holes of the barrier. This gives a simple enough model to predict the shape of the interference pattern forming on the canvas. We would like to validate the distance between the stripes in the interference pattern. To calculate the distance between the probability density peaks, first, we have to get a formula for the intensity on the canvas.

$$I(\vec{r}) = I_0 \frac{sin^2\left(\pi N \frac{dy}{\lambda L}\right)}{sin^2\left(\pi \frac{dy}{\lambda L}\right)} \tag{32}$$

where N is the number of holes on the barrier (in double-slit experiment 2), d is the distance between the center of the holes, L is the distance between the canvas and the barrier λ is the wavelength of the wave and y is the location on the canvas. We performed the simulation using $L = 30$ Bohr radii, $d = 4.0$ Bohr radii, $\lambda = \frac{2\pi}{3} \simeq 2.1$ Bohr radii with two slits. The width of each slit was a small enough value of 1.0 Bohr radii. Different stages of double-slit simulation can be seen in figure 7, where we have used ray tracing to visualize the probability density and the potential. The interference pattern is visualized in figure 8 on a canvas of size 60×60 Bohr radius. We compared the simulated pattern and the pattern predicted by the Huygens–Fresnel principle. It can be seen that the two patterns are very similar. The animation of this simulation can be accessed on https://www.youtube.com/watch?v=16M21MPFea0.

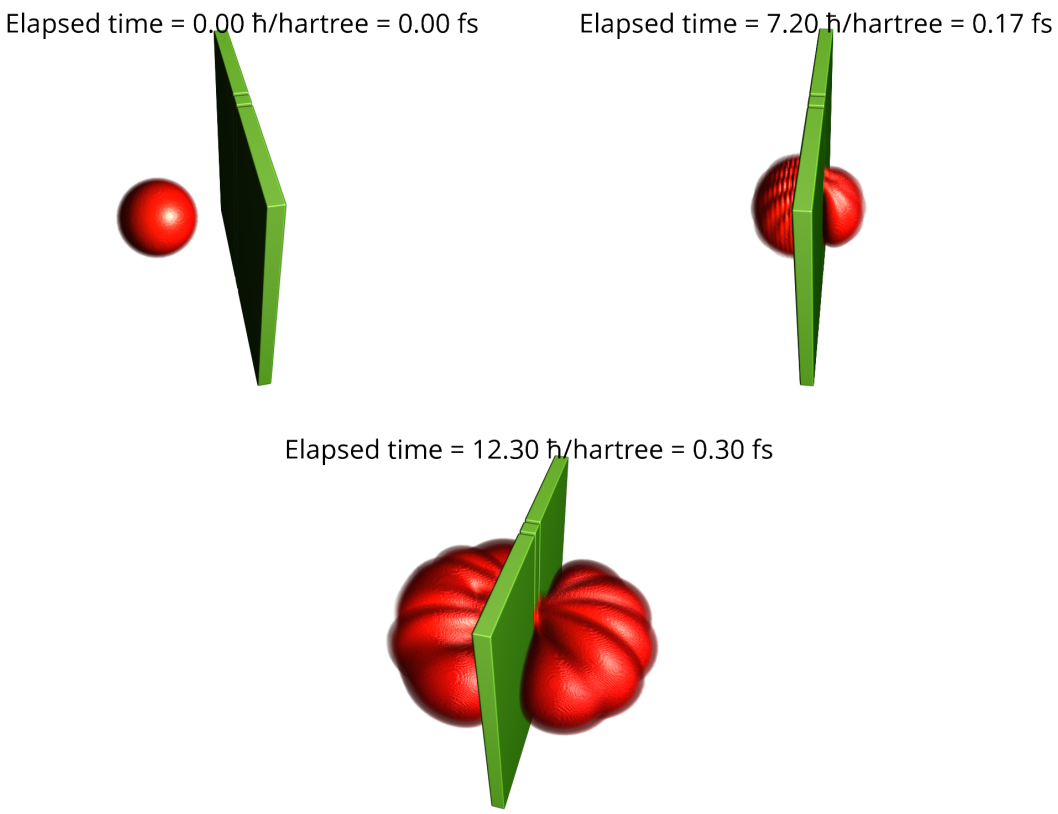


Figure 7: Stages of the double-slit experiment: before, during, and after passing through the double-slit

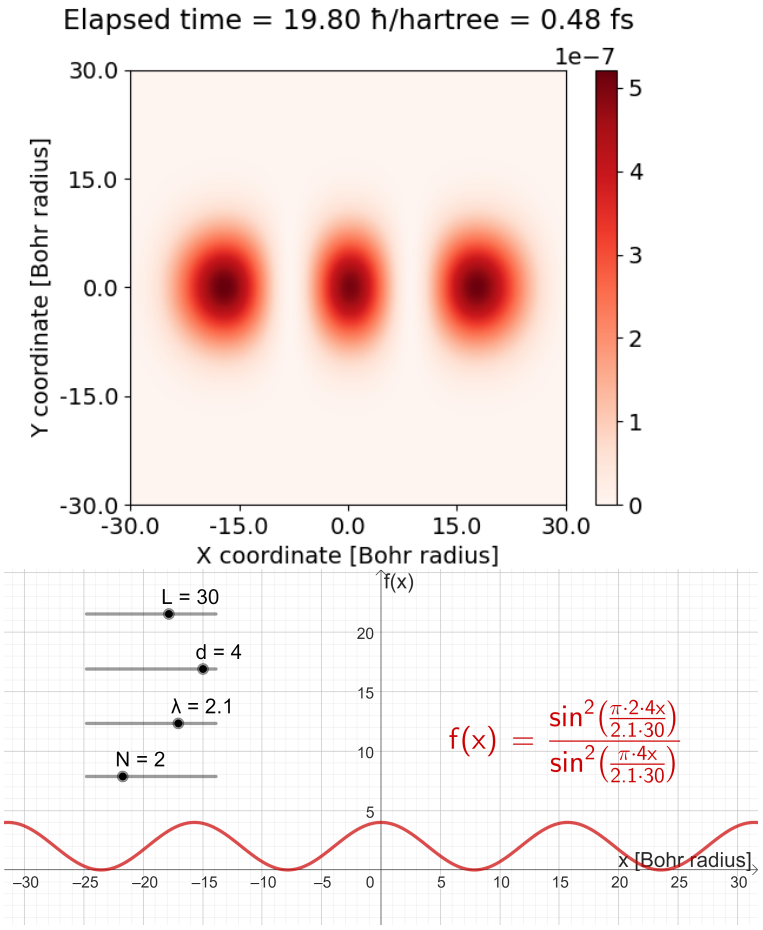


Figure 8: Comparison of the simulated (above) and the analytically predicted (below) interference pattern during double-slit experiment

5.3 Diffraction by optical grating-like potential

Many different forms of diffraction can be explained using QM. The scale at which diffraction happens ranges from the scale of subatomic particles to larger molecules. Measuring diffraction patterns is a handy tool in the hands of scientists. It provides information about the object that caused the diffraction. At this point, it is essential to emphasize that the diffraction of light is a different phenomenon from the diffraction of matter waves since light is an electromagnetic wave. With that said, the fact that particles are not electromagnetic waves but still exhibit wave-like behavior, as discussed in Section 2 makes this experiment even more interesting. Note that in equation 32, the y is a one-dimensional coordinate. Indeed, the double-slit experiment can be fully described using only two spatial dimensions, and the diffraction pattern forms in a single dimension perpendicular to the propagation direction of the wave. This is because the localized potential in this scenario is independent of the z coordinate. To make use of all three simulated dimensions, we also modeled diffraction on diffraction gratings. In optics, a diffraction grating is a periodic 2D structure that diffracts light [37]. In QM, such gratings can also be utilized to diffract wave packets. The holes between the potential nodes behave like the holes in the double-slit experiment. We put 11 nodes in each direction, forming a rectangular grid. We experimented with various lattice constants (distance between adjacent grid points). Here, we present two simulations. In the first simulation, each node has a Gaussian potential distribution and a maximal potential of $V_{max} = 8$ hartree. The distance between adjacent grid points is $d = 4$ Bohr radii. In the second simulation, each node has a Gaussian potential distribution and a maximal potential of $V_{max} = 25$ hartree. The distance between adjacent grid points is $d = 8$ Bohr radii. The canvas distance is $L = 30$ Bohr radii, and the wavelength of the WP is $\lambda \simeq 2.1$ Bohr radii for both simulations. Note in both cases the kinetic energy of the WP $E = \frac{p^2}{2m} = \frac{h^2}{2\lambda^2} \simeq 4.5$ hartree is less than V_{max}. Otherwise, the grating would not impact the propagation of the WP sufficiently. In figure 9, we visualized different stages of the time development of the first simulation, and 10 showcases stages of the second simulation. Both sequences of rendered images make use of the ray tracing visualization method.

During the simulation, many interesting interference patterns arise. We showcase some of these for the 4 Bohr radius lattice constant case in figure 11 and for the 8 Bohr radius lattice constant case in figure 12. The animations showing the results in motion can be found on https://youtu.be/KCE5xqm-diQ?si=-kzjWvaw8zOcAz4C and https://youtu.be/YCadOpsx9y8.

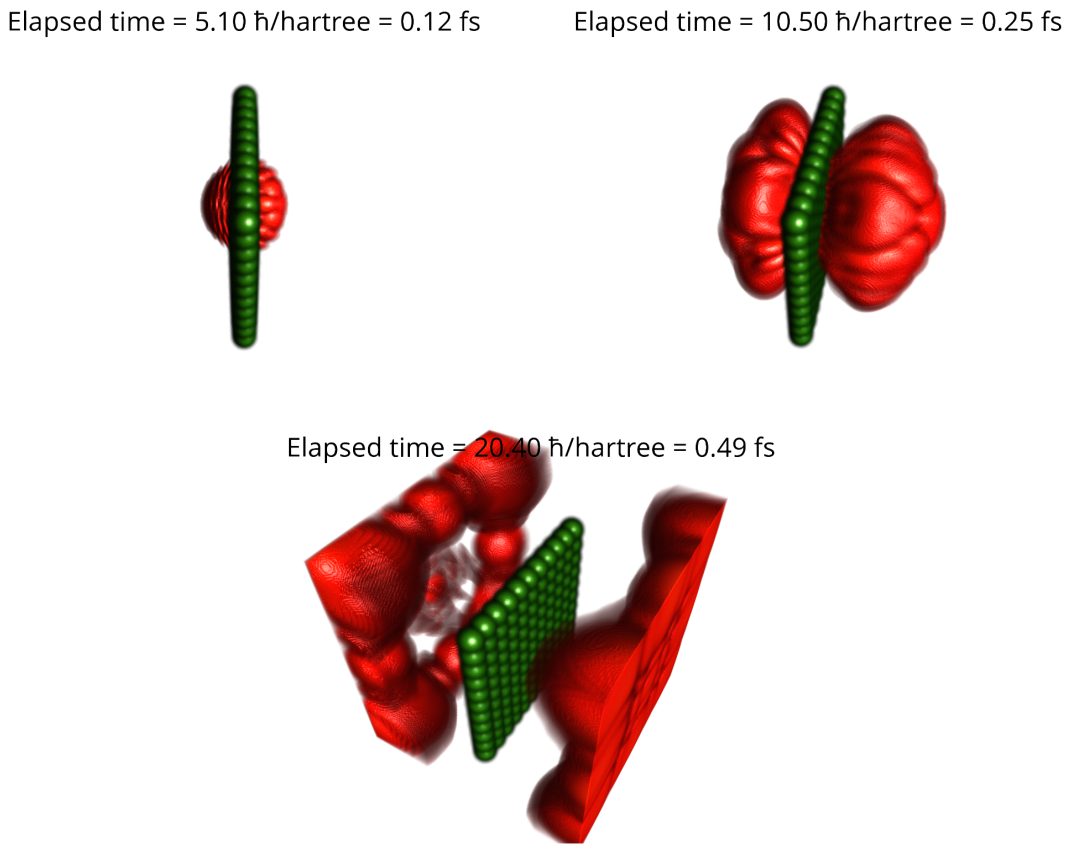


Figure 9: Stages of the diffraction grating experiment using a grating with lattice constants of 4 Bohr radii: during, right after, and later after passing through the grating

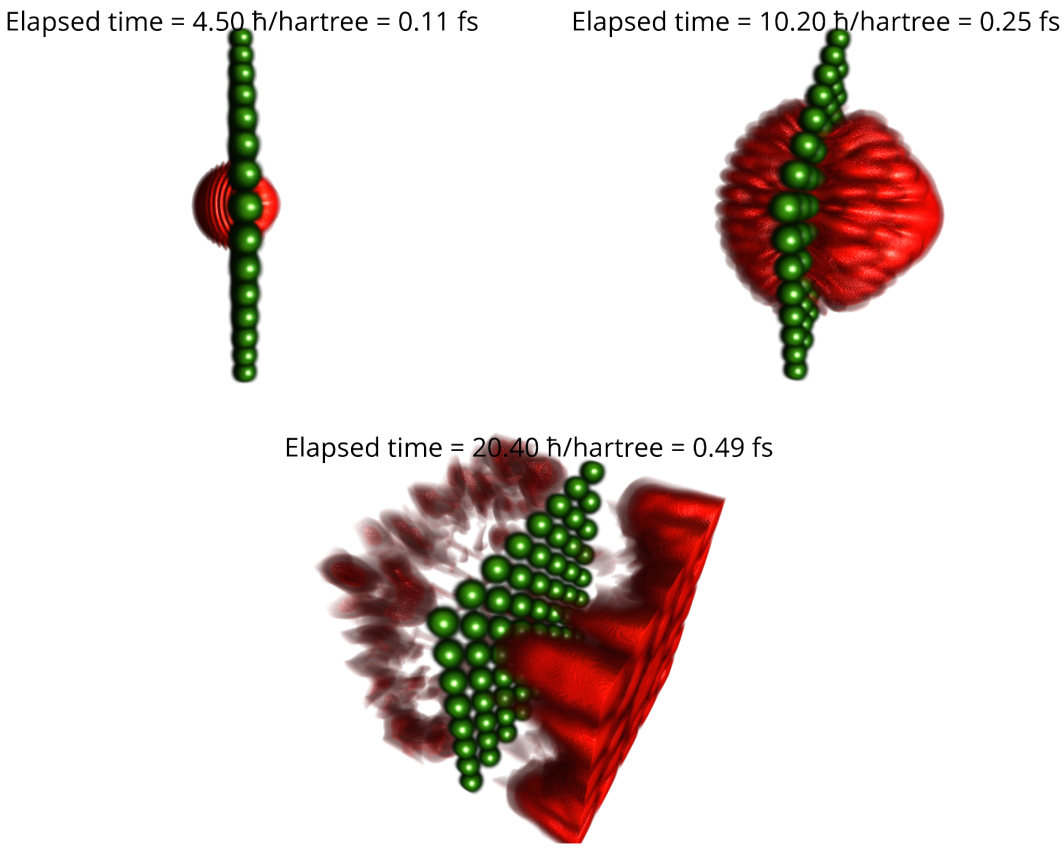


Figure 10: Stages of the diffraction grating experiment using a grating with lattice constants of 8 Bohr radii: during, right after, and later after passing through the grating

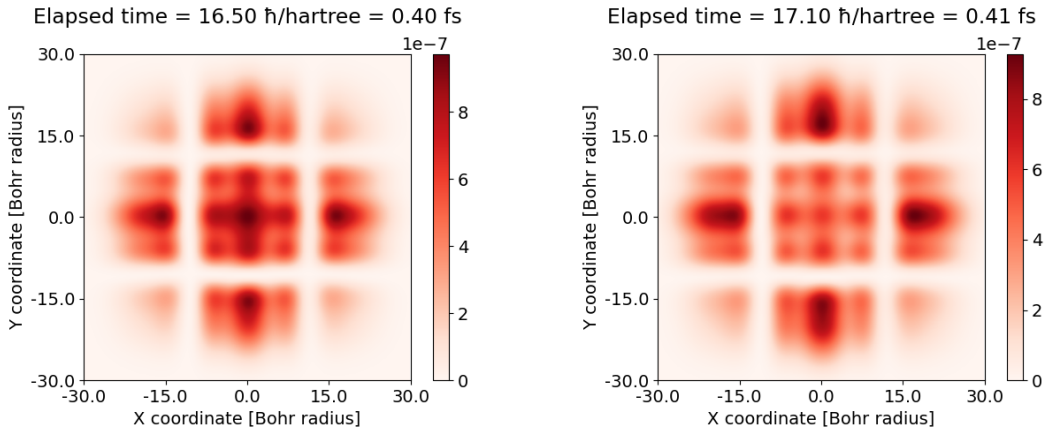


Figure 11: Two stages of interference patterns forming on the measurement canvas during diffraction grating simulation using lattice constant of 4 Bohr radii

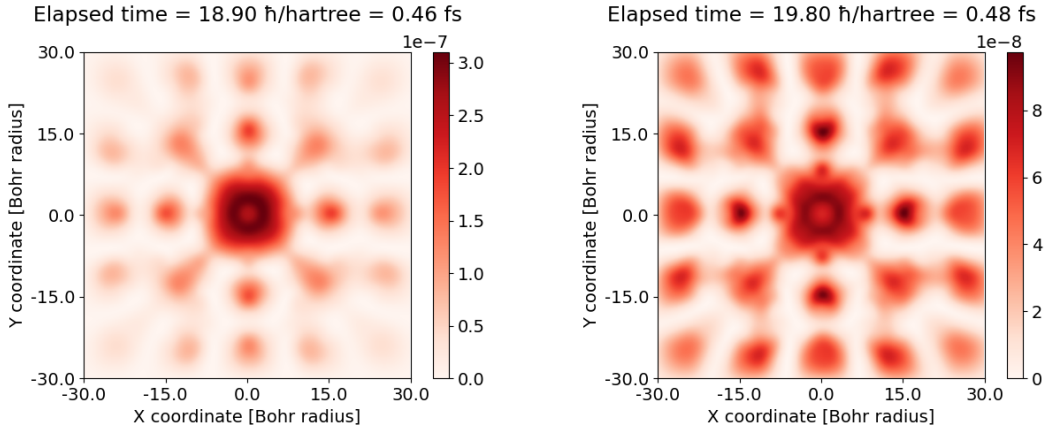


Figure 12: Two stages of interference patterns forming on the measurement canvas during diffraction grating simulation using lattice constant of 8 Bohr radii

5.4 Many-body interactions

One interesting use case of a higher dimensional WPD simulator is that the higher dimensional space can be used to model the interactions between multiple lower-dimensional particles. For example, our 3D simulation is capable of the simulation of three 1D particles. To do this, we have to define special interaction potential. To create such potential, we have to think about the coordinates in the higher dimensional configuration space as the coordinates of the lower dimensional space describing the location of the lower dimensional particle. If the potential energy affecting all particles can be expressed as a $V(x_a, x_b, x_c)$ function of the location of particle A and B and C, then we can reinterpret this function as the $V(\vec{r})$ localized potential function used in the potential propagator in equation 19. Note that here $\vec{r}$ becomes (x_a, x_b, x_c). To model the interaction between the three 1D particles, we initialized a linear combination of two types of interaction potentials. One is proportional to $\frac{1}{|x_i - x_j|}$ where $i, j \in \{a, b, c\}$ and $i \neq j$ and the other is a hard interaction that takes its maximum if the particles approach each other inside an ϵ radius otherwise it is zero. To prevent blotting of the Gaussian WP we also initialized a harmonic oscillator potential. This helps because the Gaussian WP is the eigenstate of the harmonic oscillator. The potential for a harmonic oscillator is given in the following form

$$V(x) = \frac{m\omega^2}{2}x^2 \tag{33}$$

where m is the oscillating mass and ω is the angular frequency of the oscillation. Our first simulation of 1D particles is somewhat similar to the classical Newton's cradle [38] in the sense that it demonstrates momentum transfer between multiple particles. We created a scenario where particle A starts at 25 Bohr radii away from the center of the oscillator where the potential energy is maximal; thus, it accelerates towards the other two particles (B and C), consequently transferring the momentum to particle C on the far right. For this experiment, the interaction potential consists only of the hard interaction. The angular frequency of the oscillator was selected to be $\omega = \frac{2\pi}{40} \simeq 0.1571 \ \frac{rad \cdot hartree}{\hbar}$. After the momentum transfer, particle C propagates towards the maximal potential of the oscillator on the far right. Particle B stays in place since it serves only as a medium in the momentum transfer. The momentum gained from particle A was immediately given to particle C. When all the kinetic energy of the moving particle is transferred to potential energy, then it turns back. The oscillation with the momentum transfer between the particles continues. Different phases of this simulation can be observed on the per-axis visualization in figure 13. The corresponding animation can be found on https://youtu.be/zOiYACVOssU. Please note that particle C converts all of its kinetic energy to potential energy right at the half of the first period of the oscillation $t = 40/2\frac{\hbar}{hartree} = 20\frac{\hbar}{hartree}$. This means that the harmonic oscillator functions correctly.

Now, let us repeat this experiment, but now we place a finite potential barrier in the middle of the oscillator. In Tamás Geszti's book [5], we can find that the probability of a particle with E

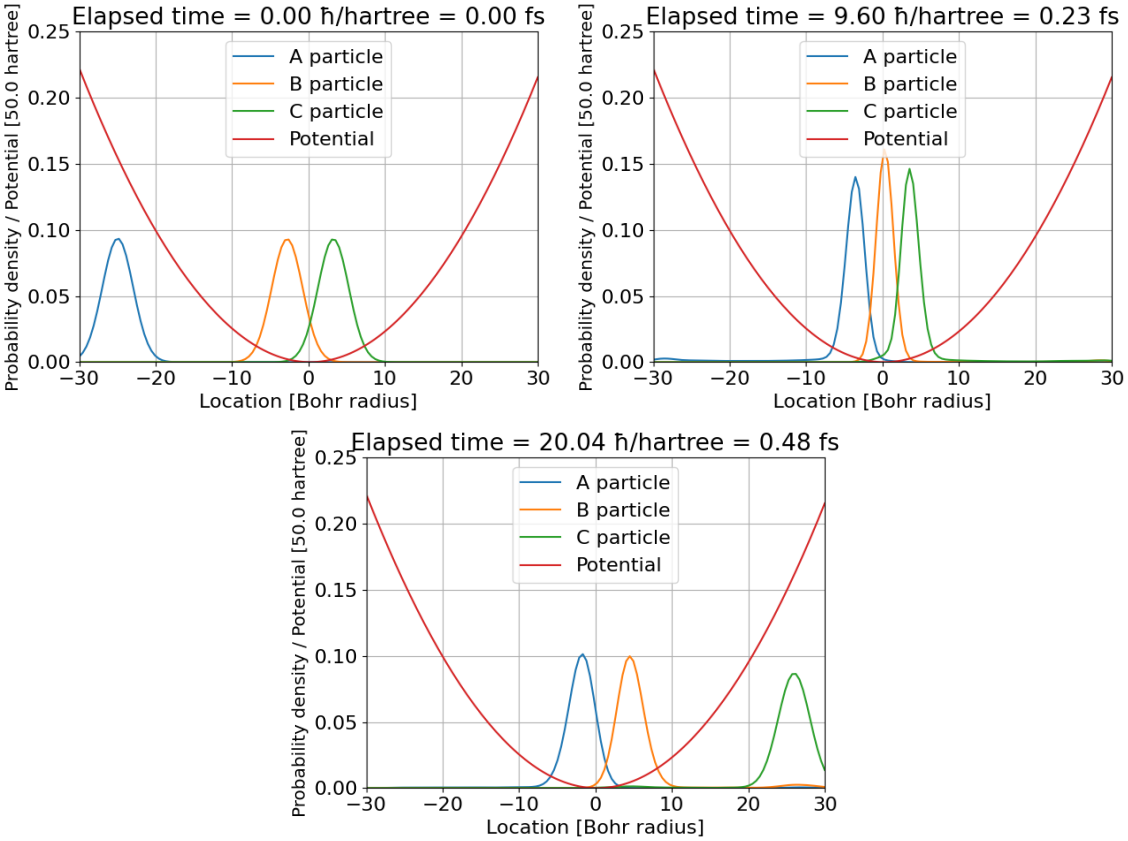


Figure 13: Stages of the interactions between 1D particles in harmonic oscillator: initial state, particle A giving momentum to particle C, particle C reaching maximal potential and turning back

energy tunneling through a d thick barrier of V_0 potential can be expressed in the following form

$$\mathbb{P}_{tunnel} = \frac{1}{1 + \frac{V_0^2}{4E(V_0 - E)} sinh^2(\kappa d)} \tag{34}$$

where $\kappa = \sqrt{2m(V_0 - E)}\ /\hbar$ is determined by the energy of the incoming WP. Energy is conserved, so $E = V(x_A)$. We solve for $\mathbb{P}_{tunnel} = \frac{1}{2}$. One possible solution is $V_0 \simeq 9.8$ hartree and $d \simeq 0.3$ Bohr radius. Particle A gives its momentum to B as before. With these parameters, approximately half of the probability density of particle B tunnels through the barrier, giving its momentum to particle B on the next side of the wall. This causes the C to start moving with a probability of approximately $\frac{1}{2}$. What we just described is called the entanglement of the states of particles A, B, and C. Let's perform a measurement to determine the location of particles A, B, and C shortly after the first momentum transfer could have happened. If we would measure particle A to be located in

the middle of the harmonic oscillator, that means that it gave its momentum to particle B and B has tunneled through the finite potential barrier. If B tunneled, that also means that beyond the barrier, it collided with particle C, consequently transferring all its kinetic energy to C. On the contrary, if the result of the measurement determining the location of particle A would have shown that particle A bounced back from B, that means that B did not tunnel through the barrier. This also means that particle C did not receive any kinetic energy and stayed stationary right beyond the barrier. The measurement of the state of one entangled particle determines the outcome of the measurement of the other entangled particles. Real-life experiments are sound with this thought experiment [39]. The probability density plot can be observed in figure 14. The animation of this simulation can be found on https://youtu.be/IVb_mpzmuYA.

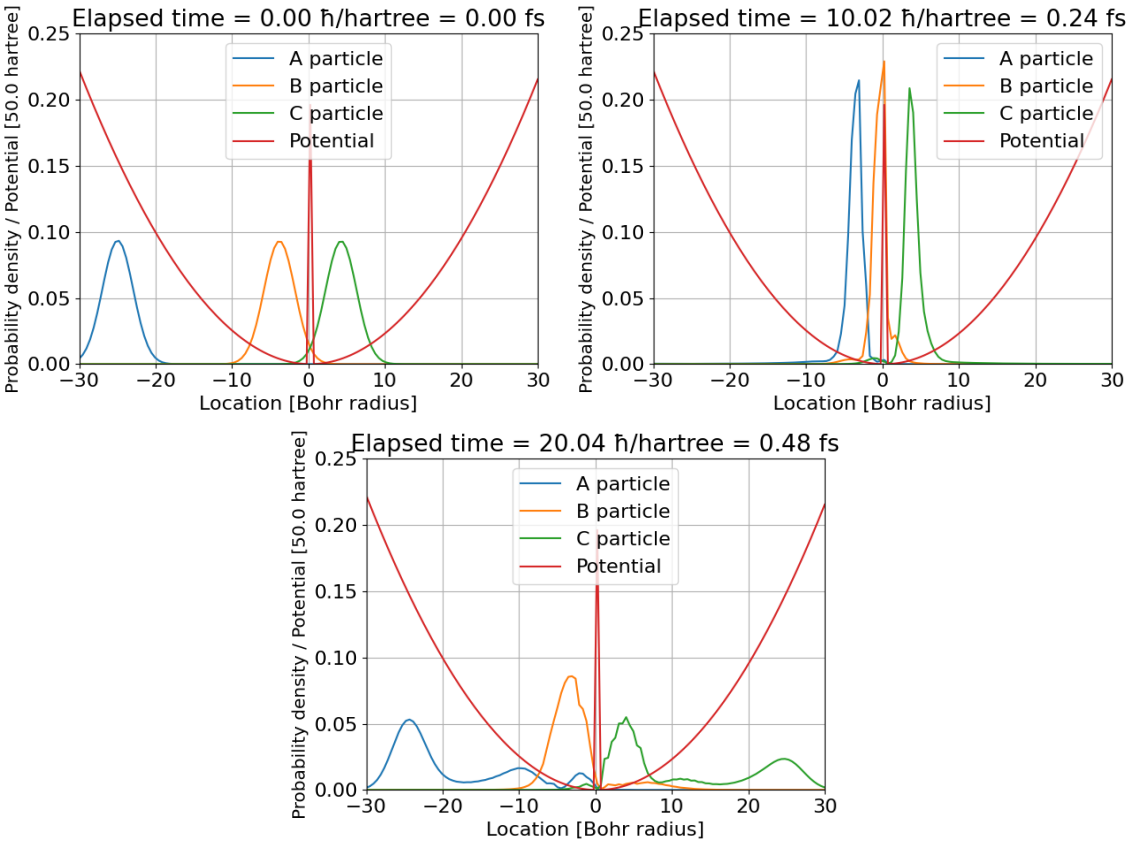


Figure 14: Stages of the interactions between 1D particles in harmonic oscillator with finite potential barrier: initial state, particle A giving momentum to particle C with approximately 0.5 probability, particle C reaching maximal potential and turning back with approximately 0.5 probability

5.5 Modeling a flash memory cell

Our last group of simulations tries to model the working of a flash memory cell. Flash memory is used extensively in many modern electronic devices such as smartphones, laptops, or tablets. The general structure of a cell is visualized in figure 15. The structure of a flash memory cell resembles a Metal–oxide–semiconductor field-effect transistor (MOSFET) [40]. The difference is that the memory cell has two gates: the control gate, which is connected to a control line similarly to regular transistors, and the floating gate, which is insulated from all sides by oxide layers. Both the control gate and the floating gate are made out of conducting material (metal or polysilicon). In figure 15, it can be seen that between the control gate and the floating gate, there is a thicker potential barrier. This prevents the flow of electrons between these two parts. Only the Coulomb potential induced by the electrons on the control gate must be felt by the electrons on the floating gate. A flash memory cell stores information by trapping electrons on a floating gate. The reading of a memory cell is done by inducing a current through the channel between the source and drain. If there are electrons trapped on the floating gate, they create an electric field affecting the substrate. The substrate is made out of semiconducting material. (Generally silicon.) A semiconductor changes conductivity based on the electromagnetic field. If there are electrons on the floating gate, the channel between the source and drain has higher resistance. By measuring the current through this channel, the information about the state of the floating gate can be retrieved. This is used to store bits of information. In our model, we simulate one single cell, and this cell only stores a single bit of information. We only model one single electron that can be trapped on the floating gate. The writing and flushing of the flash cell are performed by applying voltage on the control gate, inducing a strong enough electric field that the electron can tunnel through the potential barrier between the floating gate and the channel. For the writing and flushing to be possible,

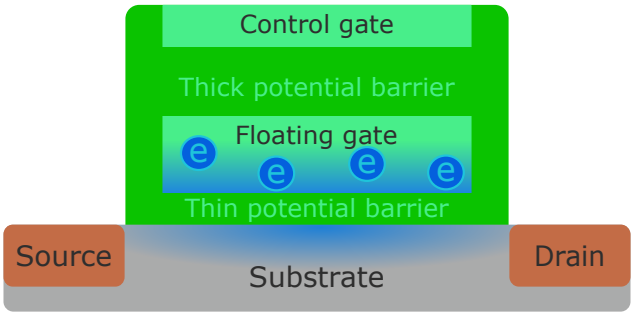


Figure 15: Schematic structure of a flash memory cell

the thickness and height of the oxide layer –consequently, the potential wall– between the floating gate and the substrate must be chosen carefully. This barrier must be thick enough to prevent unwanted tunneling when no voltage is applied to the control gate. After the control gate fills with electrons, the Coulomb potential created by those electrons makes an overall slope in the potential barrier created by the oxide layer. The potential level inside the floating gate does not acquire a

slope since it is made out of conducting material, hence its equipotential. This slope is added to the potential barrier, thus changing its shape. The previously rectangular barrier now acquired a triangular outline. More importantly, the d width of the barrier gets reduced. This increases the probability of the electron in the floating gate tunneling out from the gate through the thin oxide layer into the substrate. This phenomenon is called Fowler–Nordheim tunneling [41]. Figure 16 visualizes the principle of this effect. In our flash memory cell simulation, we initialized a potential

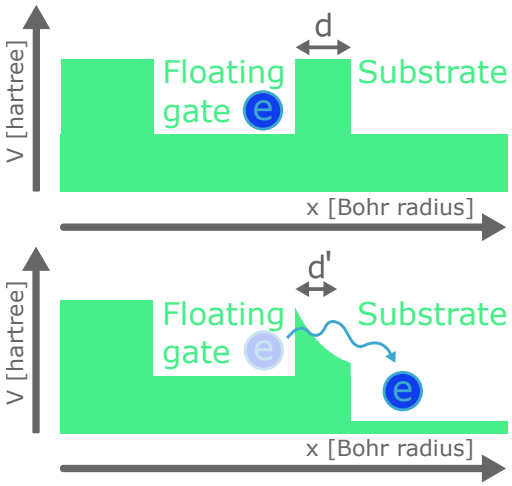


Figure 16: Fowler-Nordheim tunneling: on the upper image there is no Coulomb potential; on the image below Coulomb potential changes the shape of potential barrier causing tunneling

wall representing the barrier between the floating gate and the substrate. Our simulation deals with only one electron. To simulate the flushing of the gate, we initialize the location of this electron to be inside the floating gate. We add a Coulomb potential that decreases inside the thin oxide layer towards the substrate. We initialized the barrier with a height of $V_0 = 10$ hartree and width of $d = 4.0$ Bohr radius. We surrounded the floating gate area with thick potential barriers from the remaining five sides to prevent tunneling in unwanted directions. However, we excluded these from the visualization to make the electron's probability density visible from the outside. We model the charge on the control gate as a uniformly charged infinite sheet. This way, the Coulomb potential induced by a negatively charged control gate acting on an electron inside the oxide can be described by the following formula

$$V_C = \frac{\sigma}{2r\epsilon} \tag{35}$$

where ϵ is the dielectric permittivity of the oxide, σ is the surface charge density at the control gate, and r is the distance from the control gate. Inside the floating gate, the potential stays zero. On the thin oxide layer between the floating gate and the substrate, the slope of the potential has to be significant enough so that the probability of tunneling increases. We initialized the Coulomb potential to create a steep enough slope at the start of the oxide. For this, we set the charge density

to $600\frac{elementarycharge}{(Bohrradius)^2}$ and the thickness of the oxide between the control gate and the floating gate to 10 Bohr radius. This is significant because although the gate is on equipotential, right after the gate, the Coulomb potential continues to decrease. For simplicity, we choose ϵ to be equal to one. With this setup the derivative of potential at the start of the oxide is $V' = \frac{d}{dr}\frac{\sigma}{2r\epsilon} = -\frac{\sigma}{2r^2\epsilon} = -\frac{600}{2\cdot 10^2\cdot 1} = 3\ \frac{\text{hartree}}{\text{Bohr radius}}$. The potential barrier between the floating gate and the substrate was chosen to be 1 Bohr radius thick and 8 hartree high. An electron inside the floating gate would have some kinetic energy, so we gave it an initial momentum of 3 $\frac{\hbar}{\text{Bohr radius}}$. We ran the simulation with the Coulomb potential turned on and off and compared the *probability evolution* plots. Figure 17 shows the flushing of the floating gate with the Coulomb potential enabled. The probability of the electron being found on the floating gate is decreasing. The step-like nature of the plot is caused by the bouncing of the wave packet inside the gate. Also, note that the probability of the particle being found inside full volume is decreasing. This is because we left the underside of the substrate open to prevent unwanted reflections from that barrier. In that direction, the tunneling particle can leave the substrate freely. In our simulation, the parts of the WP leaving the visualized area get absorbed by the draining potential. At first glance, the rate at which the particle leaves the gate seems slow. Bear in mind that we are working with atomic units. Figure 17 shows that after 40 $\frac{\hbar}{\text{hartree}} \simeq 0.96$femtoseconds the probability of the particle located on the gate decreased by 10%. In figure 18, however, the electron stays on the gate

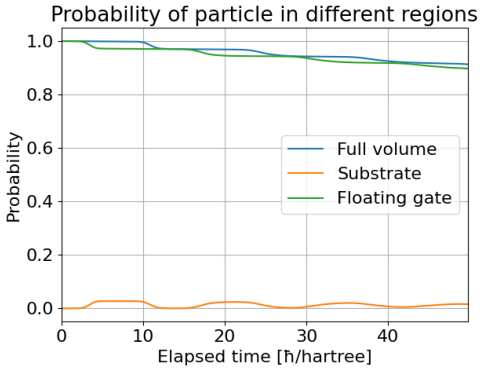


Figure 17: Flushing of the floating gate: the probability of the particle being found on the floating gate is gradually decreasing

since we disabled the Coulomb potential. For the flushing of the gate, we also provide ray-traced visualization of the flash memory cell simulation. This can be observed in figure 19 and in animated form on https://youtu.be/0-FTkSJgPPs?si=sl5k4ubo8XstPJ1b. In these images, we only visualize the potential barrier between the control gate and floating gate (top green layer), the single electron trapped inside the floating gate (with red), and the potential barrier between the floating gate and substrate (lower green layer). The simulation also incorporates the remaining walls around the floating gate and the Coulomb potential.

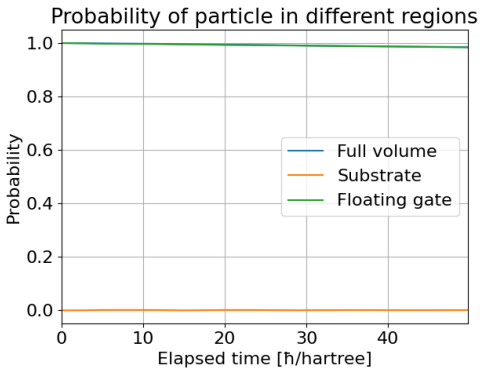


Figure 18: Storing the electron on the floating gate: there is no Fowler-Nordheim tunneling

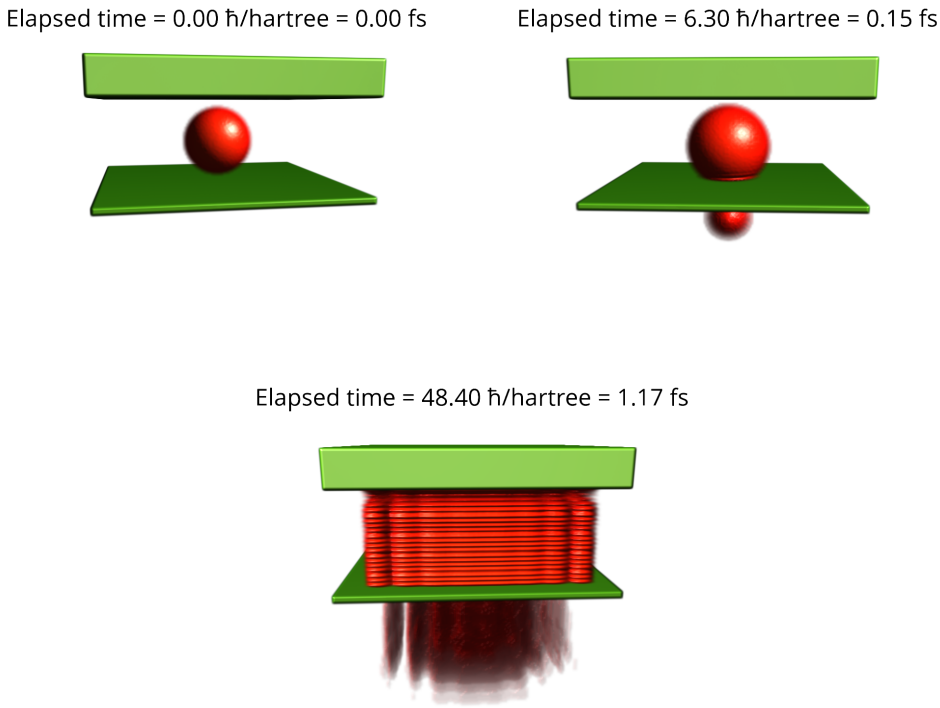


Figure 19: Flushing a trapped electron from the floating gate of a flash memory cell: initial state; first tunneling; after multiple bounces inside the gate

6 Discussion

In our work, we wrote about simulating the time development of the quantum mechanical wave function in 3D space. Our accomplishments are the following

- We adopted a simulation method that uses the Fourier transform as a subroutine to efficiently calculate the solution of the time-dependent Schrödinger equation.
- As an improvement over Géza István Márk's implementation, we ported the Fast Fourier Transform to the Graphical Programming Unit, thus reaching a major speed-up of a factor of 50 for some cases.
- We implemented the draining potential technique to allow longer simulation scenarios without the forming of unrealistic reflections and interference patterns.
- We combined state-of-the-art volume visualization techniques to enhance the visual quality of the resulting probability density images.
- We used multiple alternatives to visualize the probability densities, including *canvas probability density*, good for visualizing diffraction, the *per axis* approach that is handy when we do configuration space simulations or the *probability evolution* used in the flash memory simulation.
- We used our simulator software to run various Wave Packet Dynamical simulations ranging from basic diffraction scenarios through simulation of lower dimensional particles in configuration space up to a more advanced case of simulating the writing and flushing sequence of a flash memory cell. For some of these scenarios, we provided analytical validation methods such as the one for the interference pattern of the double-slit experiment, the easily testable periodic time of the harmonic oscillator, or even the probability of the wave packet tunneling through barriers.

We see our work as a successful entry into the world of quantum mechanical wave packet dynamics and a definitely good starting point for further research. We believe that the vast amount of application of quantum mechanics speaks for itself when the question is whether this research is important or not. Another motivation to do wave packet dynamical simulations is the Nobel prize winning research of Dr. Ferenc Krausz. Attosecond physics opens a new frontier in understanding our universe. Doing computer simulations and comparing the results with laboratory experiments can be a powerful strategy to make new discoveries. We already have multiple ideas on how to improve and append the current method. One obvious direction for development is to improve the user interface and to create a graphical interface besides the current terminal-based interface. In the future, we want to make it possible to calculate the eigenstates of the localized potential. This would require the calculation of the Fourier transform in the time domain to obtain the energy state of the system. Then, iteratively converge towards the eigenstate. There is also a possibility to incorporate electromagnetism into the Hamiltonian operator. Making this work would also be a very exciting

project. We have simulated 1D particles in configuration space. To expand on this idea, we could use higher dimensional simulations to model the interaction between multiple multidimensional particles. To simulate two 2D particles, a 4D configuration space would be required. Similarly, to simulate two 3D particles, a 6D configuration space is needed. If we would like to simulate such higher dimensional spaces, more tricks are necessary since the data sets would be too large to fit into the GPU's memory. From a visualization point of view, there are also many possibilities to improve. As seen in this work, there is room for even better reconstruction filters and clever ways to use the limited resolution. We are very hopeful about the future research potential of this topic and are very eager to continue the fruitful work.

7 Acknowledgement

I would like to thank my advisor, Dr. Balázs Csébfalvi, who has always been very supportive and open to my silly ideas. One such idea of mine was to begin to work on wave packet simulation. I am immensely grateful for the work of Dr. Géza István Márk and Dr. Péter Vancsó as my supervisors and as first-hand sources about wave packet dynamics. Without their help, this work wouldn't exist. I would like to mention Dr. János Asbóth who recommended to look into the work of Dr. Géza I. Márk's work. He gave the first quantitative impulse to this research.

Acronyms

API Application Programming Interface. 14

BRDF Bidirectional Reflectance Distribution Function. 17

CPU Central Processing Unit. 4, 14, 18

CUDA Compute Unified Device Architecture. 14

DFT Discrete Fourier transform. 10, 12

FDTD Finite Difference in Time Domain. 8

FDTD-Q FDTD technique for the analysis of quantum devices. 8, 9

FFT Fast Fourier Transform. 3, 9–12, 14

GPU Graphical Processing Unit. 2, 4, 9, 14, 15, 18, 19, 35

GR General Relativity. 3

JIT Just In Time compilation. 14

MOSFET Metal–oxide–semiconductor field-effect transistor. 30

QM Quantum Mechanics. 3–6, 11, 23

RK-HO-FDTD Runge-Kutta High-Order Finite Difference in Time Domain. 9

STM Scanning Tunneling Microscope. 13

TOML Tom's Obvious Minimal Language. 15

WP Wave Packet. 5, 11, 12, 14, 19, 20, 23, 27, 28, 32

WPD Wave Packet Dynamics. 11, 19, 27

References

[1] D. Kosloff and R. Kosloff, "A fourier method solution for the time dependent schrödinger equation as a tool in molecular dynamics," *Journal of Computational Physics*, vol. 52, no. 1, pp. 35–53, 1983. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999183900153

[2] G. I. Márk, "Web-schrödinger: Program for the interactive solution of the time dependent and stationary two dimensional (2d) schrödinger equation," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2004.10046

[3] P. Shirley, *Ray Tracing in One Weekend*, 3rd ed., 2020. [Online]. Available: https://raytracing.github.io/books/RayTracingInOneWeekend.html

[4] B. Csébfalvi, "One step further beyond trilinear interpolation and central differences: Triquadratic reconstruction and its analytic derivatives at the cost of one additional texture fetch," *Computer Graphics Forum*, vol. 42, no. 2, pp. 191–200, 2023. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14753

[5] G. Tamás, *Kvantummechanika*, ser. Elméleti fizika. Typotex, 2007. [Online]. Available: https://www.typotex.hu/book/343/geszti_tamas_kvantummechanika

[6] R. M. Wald, *General relativity*. University of Chicago press, 2010.

[7] I. Ross, "The invention of the transistor," *Proceedings of the IEEE*, vol. 86, no. 1, pp. 7–28, 1998.

[8] H. L. Zhuang and R. G. Hennig, "Computational discovery, characterization, and design of single-layer materials," *JOM*, vol. 66, no. 3, pp. 366–374, Mar 2014. [Online]. Available: https://doi.org/10.1007/s11837-014-0885-3

[9] A. K. Geim, "Graphene: Status and prospects," *Science*, vol. 324, no. 5934, pp. 1530–1534, 2009. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1158877

[10] M. D. Stoller, S. Park, Y. Zhu, J. An, and R. S. Ruoff, "Graphene-based ultracapacitors," *Nano Letters*, vol. 8, no. 10, pp. 3498–3502, 2008, pMID: 18788793. [Online]. Available: https://doi.org/10.1021/nl802558y

[11] C. N. R. Rao, K. Biswas, K. S. Subrahmanyam, and A. Govindaraj, "Graphene, the new nanocarbon," *J. Mater. Chem.*, vol. 19, pp. 2457–2469, 2009. [Online]. Available: http://dx.doi.org/10.1039/B815239J

[12] S. Imre, *Quantum Computing and Communications: An Engineering Approach*. John Wiley Sons, Ltd, 2004. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9780470869048

[13] E. Schrödinger, "An undulatory theory of the mechanics of atoms and molecules," *Phys. Rev.*, vol. 28, pp. 1049–1070, Dec 1926. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.28.1049

[14] P. Vancsó, G. I. Márk, P. Lambin, A. Mayer, Y.-S. Kim, C. Hwang, and L. P. Biró, "Electronic transport through ordered and disordered graphene grain boundaries," *Carbon*, vol. 64, pp. 101–110, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0008622313006751

[15] G. Mark, P. Vancsó, L. Biro, D. Kvashnin, L. Chernozatonskii, A. Chaves, K. Rakhimov, and P. Lambin, *Wave Packet Dynamical Calculations for Carbon Nanostructures*, 01 2016, pp. 89–102.

[16] C. Cohen-Tannoudji, B. Diu, and F. Laloë, *Quantum Mechanics, Volume 1: Basic Concepts, Tools, and Applications*. John Wiley Sons, Ltd, 2019. [Online]. Available: https://www.wiley.com/en-ie/Quantum+Mechanics%2C+Volume+1%3A+Basic+Concepts%2C+Tools%2C+and+Applications%2C+2nd+Edition-p-9783527822713

[17] W. R. Hamilton, *On a general method of expressing the paths of light, & of the planets, by the coefficients of a characteristic function*. Dublin: Printed by P.D. Hardy Dublin, 1833.

[18] J. C. Maxwell, "A dynamical theory of the electromagnetic field," *Philosophical Transactions of the Royal Society of London*, vol. 155, pp. 459–512, 1865. [Online]. Available: http://www.jstor.org/stable/108892

[19] U. Andersson, "Time-domain methods for the maxwell equations," 04 2001.

[20] A. Soriano, E. A. Navarro, J. A. Portı, and V. Such, "Analysis of the finite difference time domain technique to solve the Schrödinger equation for quantum devices," *Journal of Applied Physics*, vol. 95, no. 12, pp. 8011–8018, 06 2004. [Online]. Available: https://doi.org/10.1063/1.1753661

[21] M. Zhu, Q. Cao, and L. Zhao, "Study and analysis of a novel runge–kutta high-order finite-difference time-domain method," *IET Microwaves, Antennas & Propagation*, vol. 8, no. 12, pp. 951–958, 2014. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-map.2013.0650

[22] M. Zhu and Y. Wang, "Rk-ho-fdtd scheme for solving time-dependent schrodinger equation," *The Applied Computational Electromagnetics Society Journal (ACES)*, vol. 36, no. 08, p. 968–972, Oct. 2021. [Online]. Available: https://journals.riverpublishers.com/index.php/ACES/article/view/11753

[23] F. I. Moxley, T. Byrnes, F. Fujiwara, and W. Dai, "A generalized finite-difference time-domain quantum method for the n-body interacting hamiltonian," *Computer Physics Communications*, vol. 183, no. 11, pp. 2434–2440, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S001046551200224X

[24] J. A. Fleck, J. R. Morris, and M. D. Feit, "Time-dependent propagation of high energy laser beams through the atmosphere," *Applied physics*, vol. 10, no. 2, pp. 129–160, Jun 1976. [Online]. Available: https://doi.org/10.1007/BF00896333

[25] M. Feit, J. Fleck, and A. Steiger, "Solution of the schrödinger equation by a spectral method," *Journal of Computational Physics*, vol. 47, no. 3, pp. 412–433, 1982. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999182900912

[26] X. Zhang, *Gaussian Distribution*. Boston, MA: Springer US, 2010, pp. 425–428. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_323

[27] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[28] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[29] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson,

K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[30] (2023) Vispy. [Online]. Available: https://zenodo.org/record/4321173

[31] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," in *ACM SIGGRAPH 2008 Classes*, ser. SIGGRAPH '08. New York, NY, USA: Association for Computing Machinery, 2008. [Online]. Available: https://doi.org/10.1145/1401132.1401152

[32] R. Okuta, Y. Unno, D. Nishino, S. Hido, and C. Loomis, "Cupy: A numpy-compatible library for nvidia gpu calculations," in *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, 2017. [Online]. Available: http://learningsys.org/nips17/assets/papers/paper_16.pdf

[33] T. Preston-Werner. Tom's obvious minimal language. [Online]. Available: https://toml.io/en/

[34] J. F. Blinn, "Models of light reflection for computer synthesized pictures," in *Proceedings of the 4th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '77. New York, NY, USA: Association for Computing Machinery, 1977, p. 192–198. [Online]. Available: https://doi.org/10.1145/563858.563893

[35] R. L. Cook and K. E. Torrance, "A reflectance model for computer graphics," *ACM Trans. Graph.*, vol. 1, no. 1, p. 7–24, jan 1982. [Online]. Available: https://doi.org/10.1145/357290.357293

[36] D. R. Hartree, "The wave mechanics of an atom with a non-coulomb central field. part i. theory and methods," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 24, no. 1, p. 89–110, 1928.

[37] G. W. Stroke, *Diffraction Gratings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1967, pp. 426–754. [Online]. Available: https://doi.org/10.1007/978-3-642-46077-7_8

[38] C. F. Gauld, "Newton's cradle in physics education," *Science & Education*, vol. 15, no. 6, pp. 597–617, Aug 2006. [Online]. Available: https://doi.org/10.1007/s11191-005-4785-3

[39] D. Vasilyev, F. O. Schumann, F. Giebels, H. Gollisch, J. Kirschner, and R. Feder, "Spin-entanglement between two freely propagating electrons: Experiment and theory," *Phys. Rev. B*, vol. 95, p. 115134, Mar 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.95.115134

[40] J. Korec, *MOSFET Basics*. New York, NY: Springer New York, 2011, pp. 1–8. [Online]. Available: https://doi.org/10.1007/978-1-4419-9320-5_1

[41] R. H. Fowler and L. Nordheim, "Electron emission in intense electric fields," *Proc. Roy. Soc. Lond. A*, vol. 119, pp. 173–181, 1928.