Technische Universiteit
Eindhoven
University of Technology

Department of Biomedical Engineering
Computational Biology Group
Postbus 513, 5600 MB Eindhoven
The Netherlands
www.tue.nl

Author
Weizhou Xing

Supervisor
P.A.J. Hilbers, B.J.T. Breuer

Date
May 23, 2019

Modelling Ultrasound Propagation in HIFU System Using Beam Spreading

Internship Report

Weizhou Xing
w.xing@student.tue.nl

Where innovation starts

Technische Universiteit
Eindhoven
University of Technology

Table of contents

Title
Modelling Ultrasound Propagation in HIFU System Using Beam Spreading

1 Introduction

High Intensity Focused Ultrasound (HIFU) is a technique that uses non-ionizing ultrasonic waves to heat tissue the in human body. In the past two decades, it has been applied in treatment of a multitude of pathologic conditions. By applying high-energy ultrasound beams focused only at the region of interest, HIFU can deliver heat to the target without harming the surrounding tissue [7]. Although the therapeutic application of ultrasound is less prevalent than the diagnostic application, it is rapidly gaining popularity and is at various stages of development and commercialization [5]. The clinical and preclinical application of HIFU includes the thermal ablation of benign and maglinant lesions, targeted drug delivery through thermal-sensitive liposomes, the treatment of sonothrombolysis and so on. The simulation model in this study only focuses on the thermal ablation of bone metastases, which causes chronic and severe pain to patients and hence lower their life quality significantly. As a result, the media taken into consideration is only oil, muscle (modeled as fluid) and bone (solid). HIFU can remove the tissue that causes pain through high pressure on nerves. Clinical trials have already shown promising results, where the patient's pain is immediately relieved after treatment with HIFU[13].

When applying HIFU in therapy, doctors need visual guidance to help planning, controlling and monitoring the treatment process and to ensure the safety and efficacy of ultrasound exposure. Diagnostic Ultrasound-guided High Intensity Focused Ultrasound (USgHIFU) and Magnetic Resonance-guided High Intensity Focused Ultrasound (MR-HIFU) (Figure 1.1), are designed to provide the doctors with insightful information during the treatment process in nearly realtime. In this study, we focus only on MR-HIFU, since USgHIFU's signal to temperature mapping encounters a bifurcation for high temperature values MR-HIFU is the preferred method of temperature monitoring. In an MR-HIFU treatment, the MR-thermometry is to provide the temperature information for the doctors to plan the treatment. However, this technique is limited when bone tissue is involved. Magnetic Resonance performs better in the water-rich tissues which have a longer traversal relaxation time (T2) [11]. The concentration of water in bone is much lower than that of skin or muscle (22% compared with 75%). As a result, it has a shorter T2 and is harder to measure with high precision. What's making the problem even worse is that the temperature increase in bone tissue caused by ultrasound is faster than muscle tissue due to the different heat capacity.

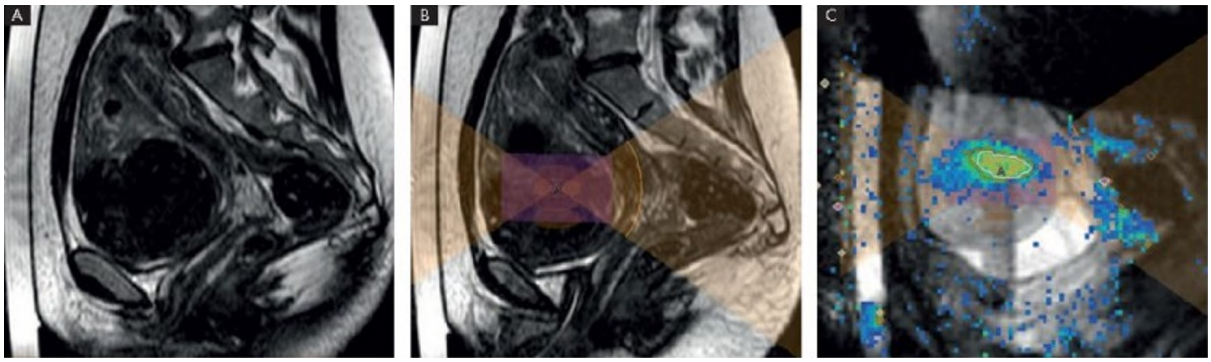


Figure 1.1: (A): an MRI-image of a tumorous uterus, (B): treatment planning for HIFU. The yellow is where the ultrasound is expected to pass, the orange is the ablation region, the purple is the target for ablation. (C) Real time temperature monitoring by MR-HIFU. The color indicates the increases in temperature. [13]

To solve this problem, computer simulation is introduced to help doctors estimate the temperature in bone tissue. Many methods have been developed to simulate HIFU systems [9]. Most of them use a ray tracing method, which treats the ultrasonic wave as ray beams propagating towards various media. However, these types of methods are usually time consuming as they require the ray tracer to cast a large number of rays so that the sampled intensity can converge to realistic values. In this report, a new ray tracer is proposed which keeps track of the divergence of the beam to estimate the intensity. The result shows that this new method can reduce the number of required rays and save some computational time.

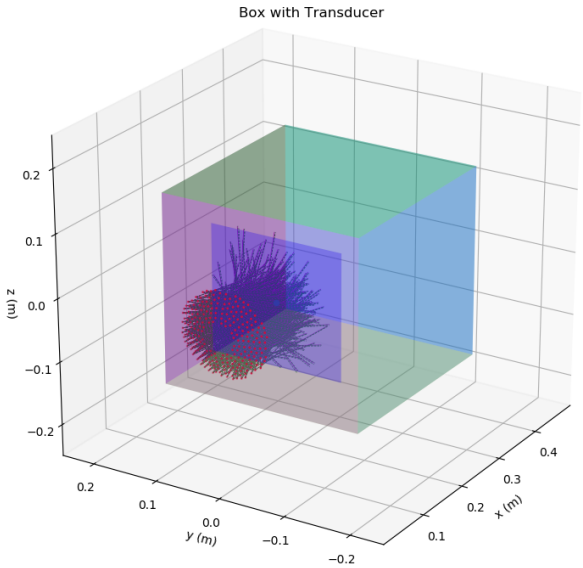


Figure 1.2: Illustration of transducer elements casting ultrasound rays at a medium interface

The study is divided into the following steps:

1. Implement a new ray tracing method using trident rays
2. Simulate a HIFU system with only one medium.
3. Simulate a HIFU system with two fluid media.
4. Parameter analysis of stem 3.
5. Qualitative comparison of the new result to the old method's result.

In the Methods section, a more detailed discussion about the theory and implementation will be presented. In the Results section, the output of the model in this study will be compared with the model of Modena et al [11], based on similarity metrics.

2 Methods

2.1 Trident Ray Theory

Currently, there are many existing ray tracing algorithms for High-Intensity Focused Ultrasound simulation [9]. Most of them aim at computing the powerloss density (heat production) or intensity from beams of rays cast from a transducer. Two existing methods will be introduced briefly because they are the inspiration of the method used in this study.

The first method is to sum over all rays (set Φ) that pass through a grid cube. Their energy loss in the cube is:

$$H = \sum_{a\in\Phi}(P(r_{a,in}) - P(r_{a,out})) = \sum_{a\in\Phi} P(r_{a,in})(1 - e^{-2\alpha||r_{a,in}-r_{a,out}||}) \tag{2.1}$$

Where H is the total powerloss by all rays in the sampling grid, α is the attenuation factor. $r_{a,in}$, $r_{a,out}$ are the entry point and exit point of the box on the ray, respectively. To obtain the powerloss density Q (in $watt/m^3$), H needs to be divided by volume. For each ray in Φ, the algorithm needs to calculate the position where it enters and leaves a gridbox. This method cannot handle the situation where intersections are present.

In the second method, the rays are grouped into paths. Note that here same path only means that the rays start from the same transducer element, pass through the media in the same order and have the same properties such as wave type. Rays with the same path are treated as a bundle (b). Take the situation in Figure 2.1b for example, the the value of the intensity loss for all the rays that follow the same path b can be calculated from:

$$I_{loss,b}(\vec{r_c}) = \sum_{a\in b} \frac{||\vec{r}_{a,in} - \vec{r}_{a,out}||}{\Delta V} P(\vec{r}_{a,in}) \tag{2.2}$$

From intensity loss I_{loss}, the pressure generated by all the rays in the same path b can be calculated from:

$$p_b(\vec{r_c}) = \sqrt{2 \cdot Z \cdot I_{loss,b}(\vec{r_c}) e^{i\bar{\phi}_b}} \tag{2.3}$$

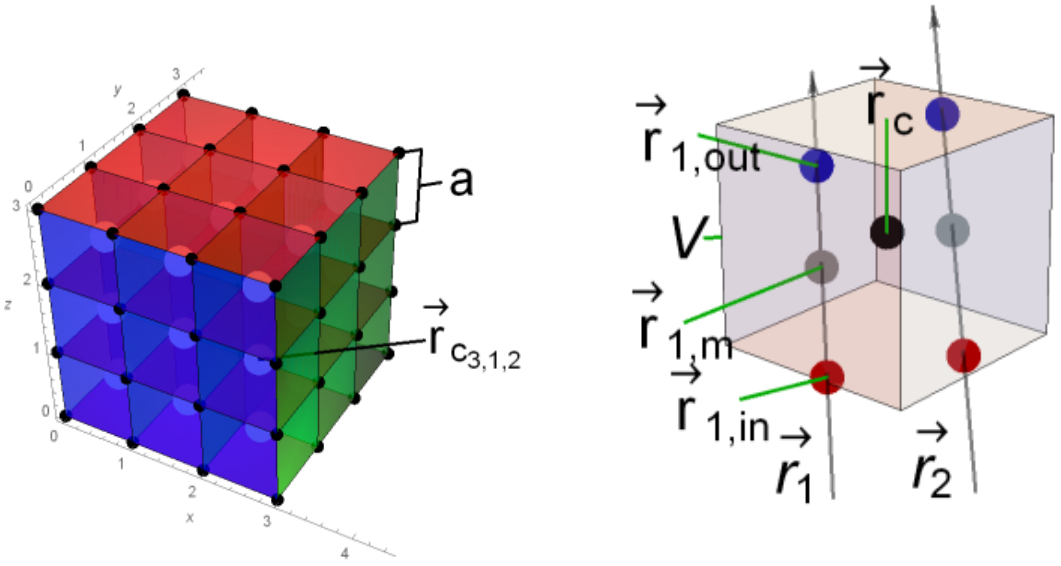
Now accumulate the contribution of all the possible bundles U to get the pressure p_{tot}:

$$p_{tot}(\vec{r_c}) = \sum_{b\in U} p_b(\vec{r_c}) \tag{2.4}$$

Accroding to Prof. Dr. Ir. Huub ten Eikelder, the fluid heat production is given by:

$$Q(\vec{r}) = \frac{\alpha}{Z}|p_{tot}|^2 \tag{2.5}$$

By combining the equations 2.2, 2.3, 2.4, 2.5, we can arrive at the equation for heat production calculated from ultrasonic rays at the center of the cube.



(a) (b)

Figure 2.1: (a) An example of Axis-Aligned Bounding Box (AABB) where $r_{c_{3,1,2}}$ is the center of the box $[3, 1, 2]$ (b) single box with intersection, r_1 enters the box at $r_{1,in}$ and exit at $r_{1,out}$, similar for r_2

$$Q(\vec{r}_c) = \frac{2\alpha}{V}\left|\sum_{b\in U}\sqrt{(\sum_{a\in U_b} P(\vec{r}_{a,in})\,\|\vec{r}_{a,in}-\vec{r}_{a,out}\|)}e^{i\bar{\phi}_b}\right|^2 \tag{2.6}$$

The flow of acoustic power from each transducer element is described by a number of rays[4]. With more rays being sent out in the simulation, the intersection of each box tends to converge to a stable ratio. As can be seem from equation 2.6,in order to obtain the powerloss density, first, the impact of all the rays are accumulated within every path (bundle). For every path, their contribution is calculated in the outer summation. In a HIFU simulation, there can be many bundles due to reflection and refraction of the rays. This will cause the computation to be very expensive when a large number of rays are utilized to approximate the real intensity.

A new idea is proposed by Prof. Dr. Ir. Huub ten Eikelder that instead of casting a large number of rays from the transducer to approximate the intensity, one can use the "trident" rays, each of which is the combination of three rays to keep track of the divergence of the beam. One of the three rays carrys the power information, the other two only serve to track the changes in area. These three rays have almost identical paths and the geometrical spreading can be tracked by the divergence of the ray pattern (Figure 2.2). At any sampling point, a plane perpendicular to the power carrying ray is created to intersect the three rays. The area of the triangle formed by the three intersection points is S_1 and can be used as how much the beam has spreaded. The power on a point along the ray is given by attenuation equation $P_1 = P_0 \cdot e^{-2\cdot\alpha\cdot r}$. Hence, the intensity and powerloss density can be derived from:

$$I = \frac{P_1}{S_1} \tag{2.7}$$

The standard way to calculate powerloss is:

$$Q = 2\alpha I \tag{2.8}$$

As the length on the ray increases, the area increases quadratically as well as the divergence of the beam. In theory, it requires only one ray per bundle to determine the intensity from this bundle. This method can achieve similar pricision with less rays, compared with the first method because it only requires that there's at least one intersection with the box per bundle.

However, in real simulation, it seldom happens that there's only one ray from a bundle that intersects a box. The contribution of the bundle to the pressure in this box will then be the average pressure from these rays. More detailed discuss is available in section 2.6.

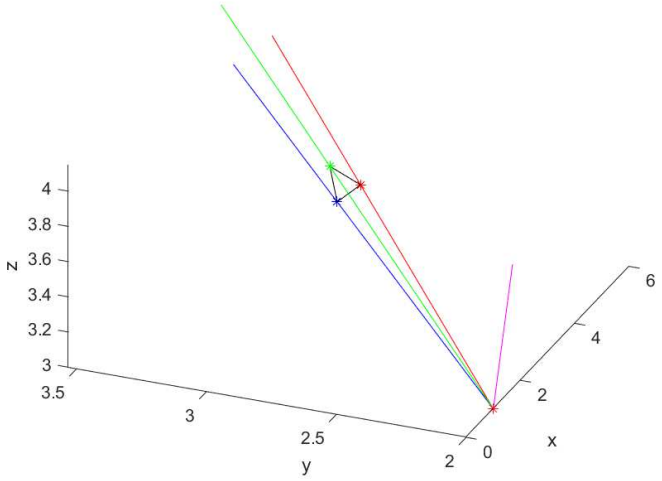


Figure 2.2: An example of trident rays, the angle between rays are larger for demonstration purpose.

2.2 Components of HIFU Model

An MR-HIFU system is a highly sophisticated system with many components. For simplicity reasons, the MR-HIFU system is modelled as being composed of only transducer, media, rays and sampling box. The model uses the actual geometry of the V2 phased array transducer of the Philips Sonalleve system [4], which consists of 256 elements placed on a spherical shell (12 cm radius of curvature and 13 cm aperture) operating at 1.2 and 1.4 MHz. The focus point created by this system is cigar shaped with dimensions of approximately 2 mm width and 7 mm length. The power of the transducer can be set as a parameter in the model, in this study, 256 watt (1 watt per transducer) is used as the total power of the transducer.

Medium	*TH*	c	ρ	α	c_p	k
Oil	-	1430	1070	1.04	4200	0.5
Gel pad	15	1580	1030	0.8	3600	0.65
Skin	2	1610	1200	50	3600	0.56
Muscle	10	1547	1050	9.8	3600	0.56
Bone	-	3736/1995	2025	1.9/2.8	3720	0.487
Liver	-	1578	1050	5.65	3600	0.56

Table 2.1: Properties of the materials used in this study. *TH*: thickness, c: speed of sound, ρ: density, α: attenuation, c_p: specific heat capacity, k: thermal conductivity. [11] [12]

The transducer is submerged in Castor oil which has a very low attenuation factor for ultrasonic power (Figure 2.3). The acoustic property of Castor oil is given by Ramaekers et al [12]. The Castor oil is separated from the air by a thin membrane. The tissue is modeled as consisting of only bone and muscle. Skin and other soft tissues have very close properties to that of the muscle and thus are not discriminated. For ultrasonic waves, the muscle and the skin behave as a fluid and the bone behaves as a solid. By definition, propagation of shear (transversal) wave is only possible in non-fluid. Castor oil is the initial medium of the trident rays.

Figure 2.3: Transducer element submerged in Castor oil (red circle) [4] [13].

To measure the power production in the media caused by the ultrasound, a sampling box is employed (Figure 2.4). Since many sampling points are required due to the small wave length, the sampling box is kept small, only around the focus region, which would also be the lesion area in therapeutic application. The box is divided into smaller cubes. The smaller the cubes are, the finer the resolution (Figure 2.1a). The edges of the cubes should be larger than the ultrasound wave length. Two measurement methods are implemented. The first one uses the projection of the center of the cubes on the ray as the length to calculate intensity and add the intensity of all the rays together. The second method differs from the first method only in that the summation is weighted based on the length of the intersection of the ray with the cubes.

As explained in Section 2.1, a trident ray in this model consists of three closely aligned rays. One ray carrys the power and is referred to as "power ray". The other two rays only help to determine the divergence of the beam and are referred to as "auxiliary ray". The angle between any two rays of the three is referred to as "trident angle".

2.3 Implementation of the Model

The mathematical model is written in the Python programming language, using scipy suite[8]. The visualization of the model is made possible with matplotlib [6]. The model is produced as a python package `pyHIFU` and is available online. The four components mentioned in section 2.2 will be introduced in detail. A Python class is implemented for these four component. This make the program more intuitive and easy to maintain. Other than these four component, the geometry and acoustics are also implemented in an object-oriented manner. The class `HIFU` encapsulates several methods for manipulating the four components above. As is shown in Figure 2.5, `transducer` casts `ray` at `media`. The rays are grouped into bundle according to their wave type and media history. Then, `box` samples the intensity generated by every bundle to

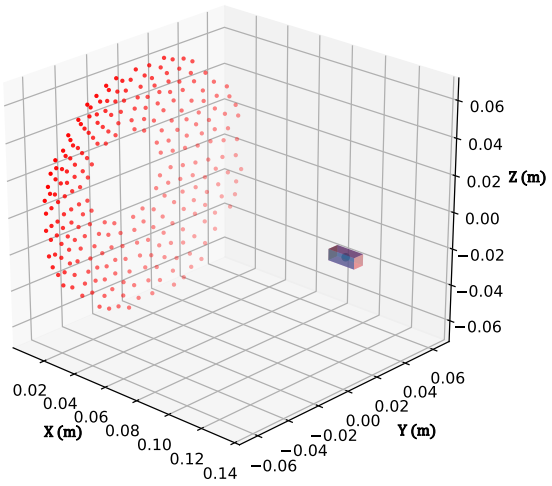


Figure 2.4: Sampling box around the focus point with positions of the transducer elements. All the figures in this report follow the axis convention here.

get the resultant pressure. The same routine is run for all the transducers. The final result is the summation of the result from all transducers. Initialization of the model requires a lot of parameters, a json document is used as configuration file to simplify passing parameters.

2.3.1 Media

A medium is implemented in class `Medium`. The medium in this study mainly consists of two parts, geometry and material. The geometry of the medium is an instance of the shape in module geometry (Appendix C) and defines the shape of the medium in 3D space. The material of the medium defines the physical properties of the medium, including all the properties in Table 2.1. For example, a medium can have the cuboid as its geometry and the bone as its material. Some properties such as the speed of sound and the attenuation in solid media have different values for longitudinal waves and shear waves. These properties would be stored as a list with the first element as the property of longitudinal waves and the second element the property of shear waves. A shape contains several faces. For example, the shape of a cylinder is composed of two circles and a barrel shape. These faces are organized as a list and each of them has its own ID. When a instance of a geometry is initialized, a matrix is created as its property to store the topological relationship of the shape. A typical such matrix of a cylinder would be

$$\begin{bmatrix} None & 0 & None \\ 0 & None & 1 \\ None & 0 & None \end{bmatrix}$$

For example, the first element of the second row is 0 because the barrel shape is adjacent to the upper circle on its first (index 0) edge. The two circles are not adjacent, thus the corresponding

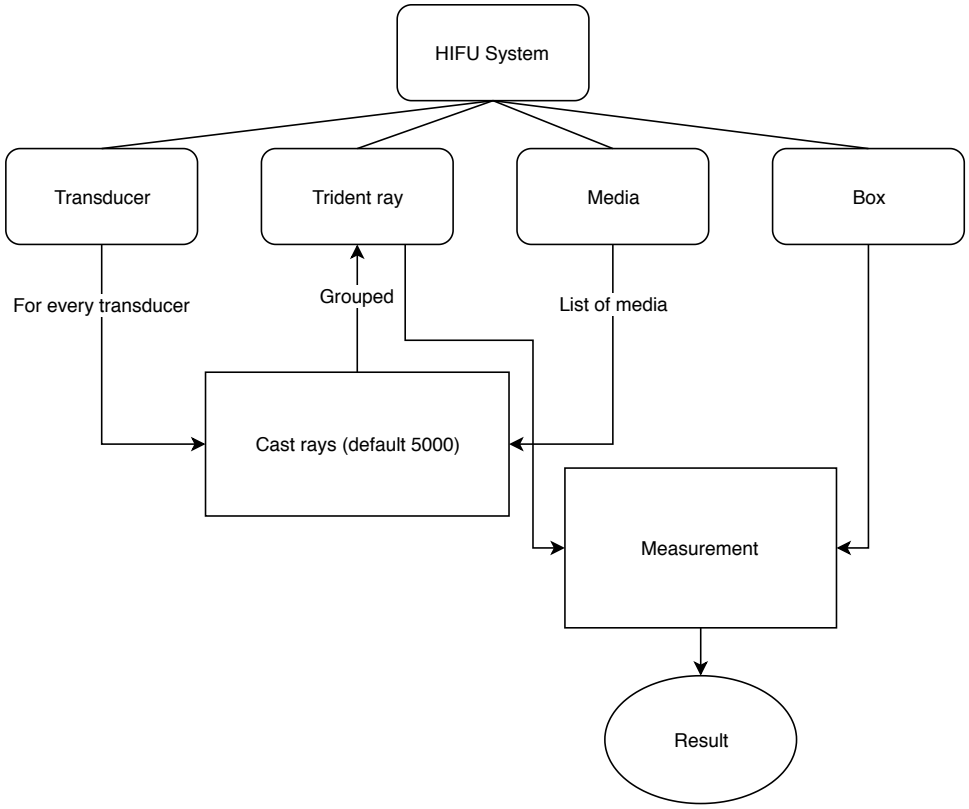


Figure 2.5: An overview of all the components in the HIFU system.

elements are $None$.

A Model can contain several media. They can be identified with their `ID`. The media are stored in a graph structure named `MediaComplex`. For example, in the simplest case, the only medium in the system is Castor oil *medium 0*. If muscle is taken into consideration, *medium 0* would be Castor oil and *medium 1* muscle. The `MediaComplex` is implemented such that the first element is always Castor oil or any other initial medium. When a ray travels in a medium, it is important to be able to determine the next medium so as to calculate the percentage of the power that is transmitted and reflected to the various ray types. The graph structure in this model is implemented as a list with an adjacency matrix `adj_mtx`. The adjacency matrix is a square matrix M. $M_{i,j}$ is the index of the face (Appendix C) of *medium i* that is adjacent to *medium j*. In this study, the media all have a cubiod shape. It can be extended to more complex shapes such as cylinders or spheres with a little effort.

2.3.2 Trident

As mentioned earlier, the ray that carries power is implemented as class `PowRay` (power ray) and the ray that only helps to determine the divergence is implemented as class `AuxRay` (auxiliary ray). They both inherit from a general purpose ray class `Ray`. Acoustic rays in this model are represented as geometrical rays `pyHIFU.geometry.Ray`. Class `Ray` is a subclass of the geometrical ray class `GeoRay` (Appendix C), adding some properties and methods for acoustics simulation.

A trident ray (class `Trident`) consists of three rays, one `PowRay` and two `AuxRays`. Every trident has a unique id. Interaction with media doesn't change the id. Member property `el_id` stores the id of the transducer where it is cast from. The trident class also contains the reference to the

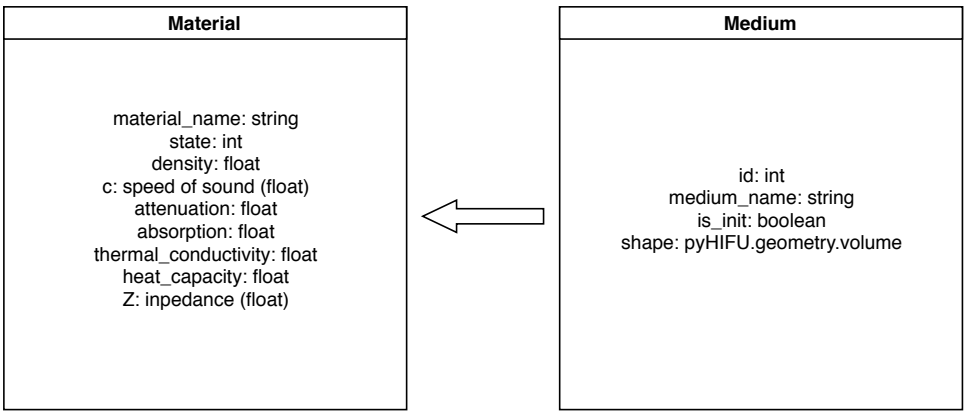


Figure 2.6: Inheritance relation between class Medium and class Material.

media that it resides in. The member property `history` records all the media that trident ray has passed through. `wavetype` is an integer variable that indicates whether the ray is longitudinal (0) or shear (1) wave. A ray can encounter several intersections with media interfaces and produce up to four child rays in case of solid-solid intersection. Each intersection splits the parent rays into reflected rays and transmitted rays, the power is divided over the rays according to the solutions to the acoustic Fresnel equations for the given situation. Once the power of the ray is lower than a predefined threshold `powerlimit`, the ray is discarded and the propagation stops. The trident rays are grouped into bundles according to their `bundle_identifier`. The `bundle_identifier` is a string that is composed of the id of the transducer the ray originates from, its history and its wavetype. For example, `tr_128_mh_0_1_1_1_LONGIT` can be intepreted as a longitudinal trident ray cast from transducer element *no.128* propagating from *media 0* into *media 1* then reflected inside *media 1* twice until its power is lower than `powerlimit`.

Ideally, the pressure profile of the ultrasound field in the far-field and near-field can be predicted using the wave equations [13]. However, the difficulty arises due to rising computational cost with increasing geometrical complexity when using the classic finite element or boundary element approach. In this simulation, only if a trident is cast from a transducer element, not generated by reflection or refraction, is its initial intensity determined by the far-field approximation of the ultrasound profile (Section 2.6). Otherwise, the initial intensity is determined using Fresnel's equations. There's a minimal distance for a far-field approximation to hold [2]. This distance is referred as D_z in this study. It is a intrinsic property of the transducer elements.

$$D_z = \frac{2.3 \cdot r_t}{\lambda} \tag{2.9}$$

r_t is the radius of the transducer element and λ is the wave length of the ultrasound. The initial phase of all rays from the transducer element is 0. When initializing a trident ray, the initial pressure is calculated from the far-field approximation at D_z. The initial intensity is calculated

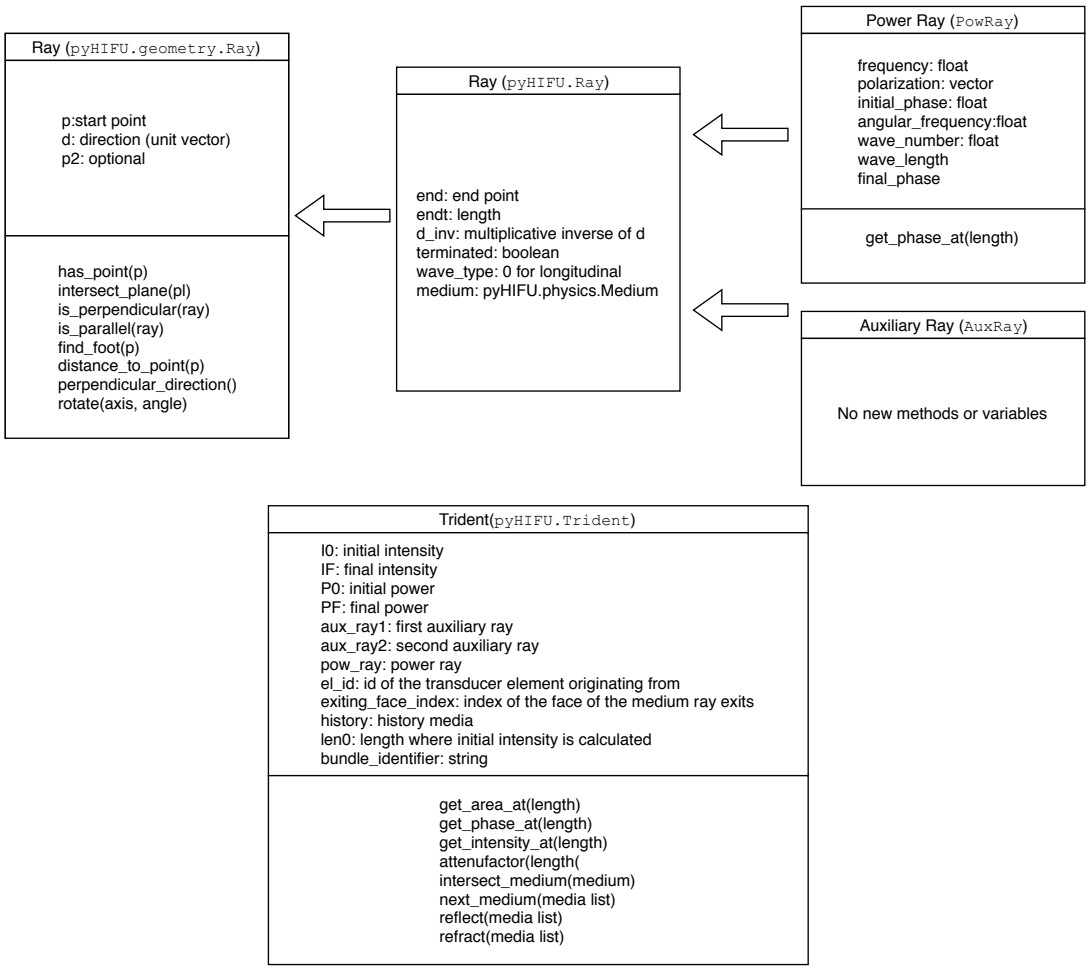


Figure 2.7: Relation among class `pyHIFU.geometry.Ray`, `Ray`, `PowRay`, `AuxRay` and `Trident`.

from equation 2.10a.

Conversion between intensity and power:

$$I_0 = \frac{pressure^2}{2 \cdot Z} \tag{2.10a}$$

$$P_0 = I_0 \cdot Area(D_z) \tag{2.10b}$$

$$P_r = P_0 \cdot e^{-2\alpha r} \tag{2.10c}$$

$$I_r = \frac{P_r}{Area(r)} \tag{2.10d}$$

After initialization, the far-field approximation is no longer used. The area formed by the three rays is affected by the solid angle (trident angle) and so is the initial power 2.10b. The area in equation 2.10d increases quadratically as the distance from the origin of the ray increases. Hence, the intensity decreases quadratically due to beam spreading and exponentially due to attenuation.

P_0 in equation 2.10b is derived from real intensity I_0 from the far-field approximation and spreading area $Area$. Equation 2.11 shows that the intensity of a trident decreases quadratically with length, with exponential attenuation. Although power is affected by trident solid angle, the intensity is not determined by the trident solid angle at all.

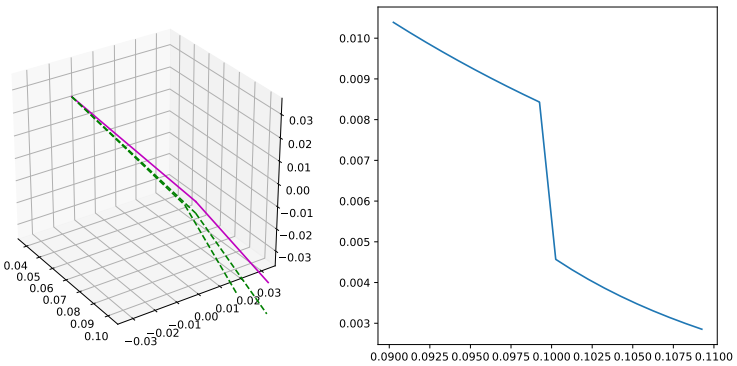


Figure 2.8: The intensity change along a trident ray with one refraction. The green dash lines are auxiliary rays and the magenta ray is the power ray. The turning point on the ray is the position of the interface (left). As can be observed in the figure, the intensity first changes exponentially, then decreases suddenly after the intersection with media interface at 0.1 m (right). After the intersection, the intensity again changes exponentially according to the attenuation equation.

$$\begin{aligned} I_r &= \frac{P_0}{Area(r)} \cdot e^{-2\alpha r} \\ &= \frac{I_0 \cdot Area(D_z)}{Area(r)} \cdot e^{-2\alpha r} \\ &= I_0 \cdot \left(\frac{D_z}{r}\right)^2 \cdot e^{-2\alpha r} \end{aligned} \tag{2.11}$$

When the ray travels inside a bounded media, it eventually encounters the boundaries of the media. To calculate the direction of new rays, some properties of the new medium are required. As explained in Section 2.3.1, the media are stored as a graph. The method `next_medium` of class `Trident` locates the next medium by looking up the adjacency matrix. A trident ray contains three rays. Each of these rays intersect with the boundaries and produces their child rays. These child rays are grouped into a new trident after intesection, i.e., reflected longitudinal power ray and reflected longitudinal auxiliary rays are grouped into a new reflected longitudinal trident (Figure 2.9). If auxiliary intersects outside the interface, the interface is expanded as an infinite plane to guarantee that the auxiliary rays and the power ray in the same trident should always intersect the same interface.

2.3.3 Transducer

A transducer element is a device that can convert other forms of energy into acoustic energy in the form of ultrasound. The transducer in this study consists of 256 transducer elements. In the model, the transducer is implemented as class the `Transducer` and the transducer element is implemented as the class `TElement`. `Transducer` maintains a list of instances of the class `TElement`, in this case, the length of the list is 256. Every transducer needs to cast rays and group them according to its bundle identifier string. This is implemented as a Python dictionary of which the keys are the bundle identifier string. As mentioned earlier, rays are discarded if their power is lower than power limit. In this model, the power limit is defined as 0.1% of the

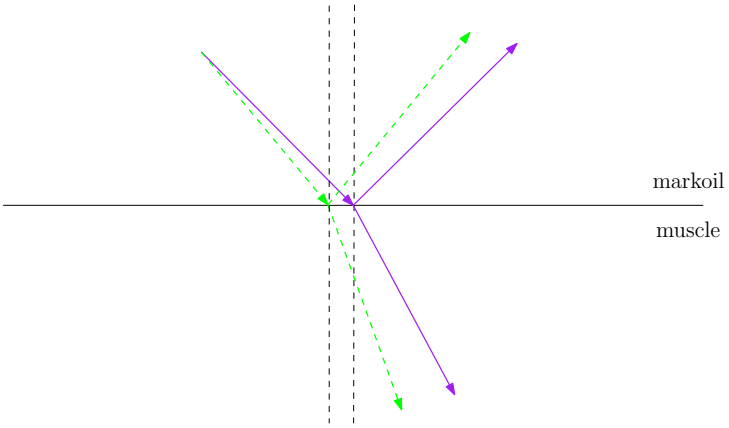


Figure 2.9: The purple ray is a power ray and the green dashed line is an auxiliary ray. For simplicity, only one auxiliary ray is displayed. Every interaction of a trident with an interface gives rise to three new rays which will make up a new trident.

initial power.

Algorithm 1: Casting rays from one transducer element

input : $nray$: number of the rays from one transducer, $trident\_angle$: angle between `AuxRay` and `PowRay`, θ_{max}: the maximum angle from the axis of a transducer element to cast rays.
output: bundles: dictonary with the bundle identifier string as the key

```
bundles ← empty dictionary
total_ray_number ← 0
for i ← 1 to nray do
    θ_i ← random angle between 0 and θ_max
    β_i ← random angle between 0 and 2π
    p_i ← point on polar coordinate defined by θ_i and β_i
    tr_i ← trident at the direction of p_i
    queue ← empty queue
    append tr_i to queue
    while queue is not empty do
        tr ← queue pop from left
        if the power of tr is less than powerlimit then
            discard tr
        else
            put tr into bundles
            append reflected rays of tr to queue
            append refracted rays of tr to queue
return bundles
```

The operation on every transducer element is independent from each other. Multiprocessing is employed to accelerate the simulation process, with each transducer being handled by one process. The result from every transducer element is summed to yield the final result (equation 2.13).

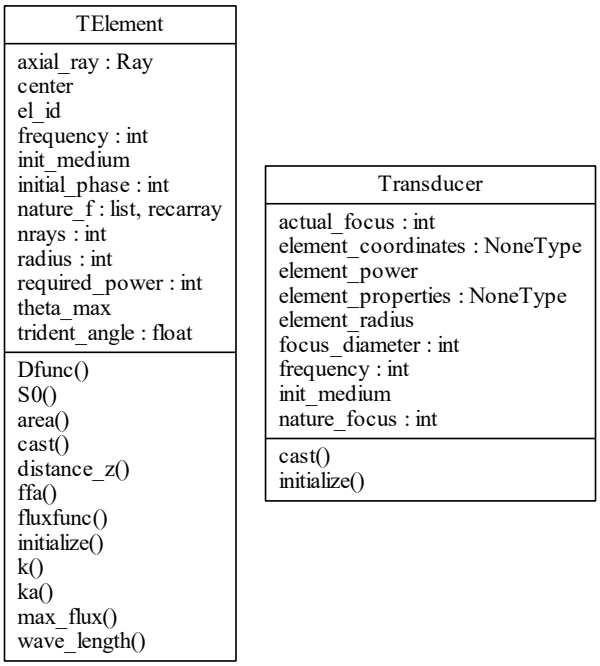


Figure 2.10: Class `pyHIFU.transducer.Transducer`, `pyHIFU.transducer.TElement`.

2.3.4 Sampling box

Only the area around focus point of the HIFU system is measured in this model. Box defines an "axis-aligned bounding box" in the geometry where the pressure will be measured (Figure 2.4). It is divided into smaller cubes implemented in the class `Lattice` (Figure 2.1a).

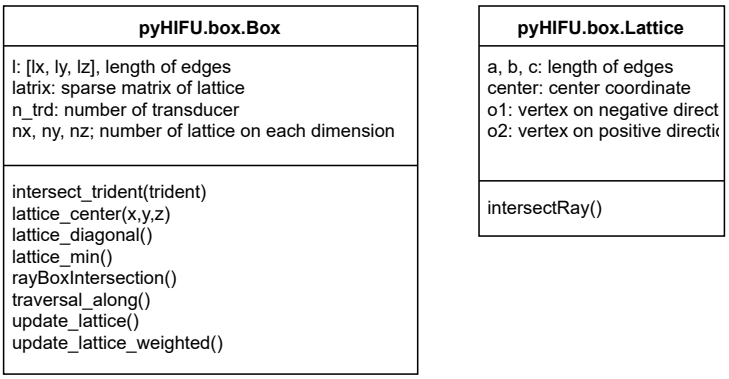


Figure 2.11: Components of the class Box and the class Lattice.

`Box` takes a function handle as its argument. When the sampling box traverses along a trident ray, it performs updates to all the intersected cubes using this function handle. In this study, the sampling method uses the ratio of the length of intersection segment and the cube diagonal as the weight for intensity contribution. In section 2.5 and 2.6, it will be further illustrated.

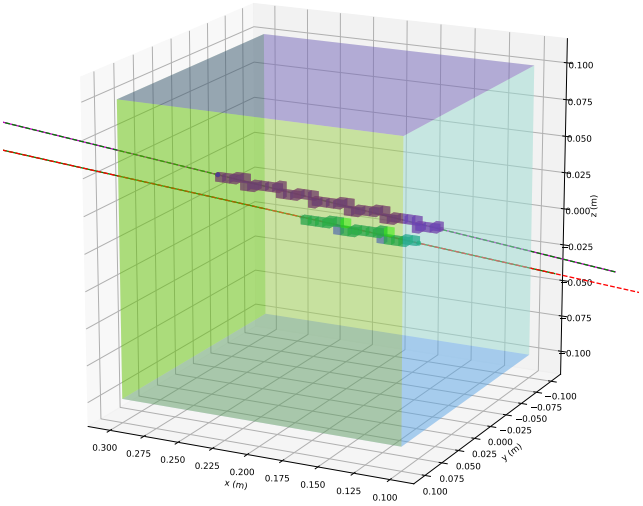


Figure 2.12: Box intersecting with two trident rays in two directions (forward and backward). The cubes that intersects the ray in different directions are displayed in different colors.

2.4 Reflection and Transmission

When a ray travels in the geometry, it encounters media interfaces and is reflected and refracted. Considering the trident ray (Section 2.3.2) in this model, we simply perform reflection and refraction on both the power ray and the auxiliary rays. Shear waves can only travel in solid media while longitudinal waves can travel in both solid and fluid media, hence resulting in four types of interactions, fluid-fluid, solid-solid, fluid-solid and solid-fluid interactions (Figure 2.13). There are two laws used in the simulation of reflection and transmission: Snell's law that calculcates the direction of new rays and Fresnel's equations that calculates the energy distribution among the rays.

Fluid-fluid interaction is the most simple one among the four types. In a fluid medium, only longitudinal waves can propagate. The reflected ray and transmitted ray are longitudinal only. Solid-solid interaction is a more complicated situation since it also needs to consider the polarization directions. In this model, only the fluid-fluid interface is implemented and tested because of its simplicity. Future work will include implementing the functionality of shear wave reflection and refractions.

For both reflection and refraction, the direction of the new rays are determined by Snell's law (Equation 2.12). θ_x refers to the angle between $\vec{n}$ and reflected rays or the angle between $-\vec{n}$ and transmitted rays as shown in Figure 2.14. If the reflected ray and the incident ray have the same wave type, i.e., they are both longitudinal or shear waves, the reflection angle θ_r is equal to the incident angle θ_i.The speed of longitudinal wave and shear wave are different in the same medium. If they have different wave types, according to equation 2.12, the two angle should also be different.

$$\frac{sin(\theta_i)}{sin(\theta_x)} = \frac{c_1}{c_2} = \eta \tag{2.12}$$

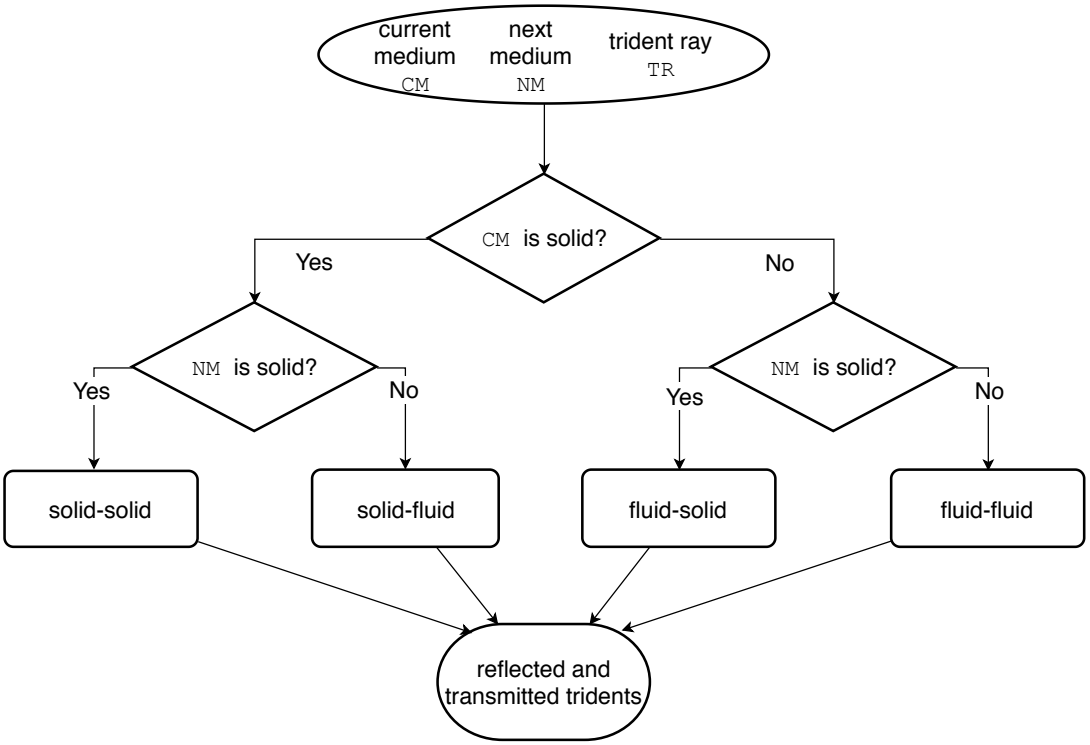


Figure 2.13: Flowchart of reflection and transmission

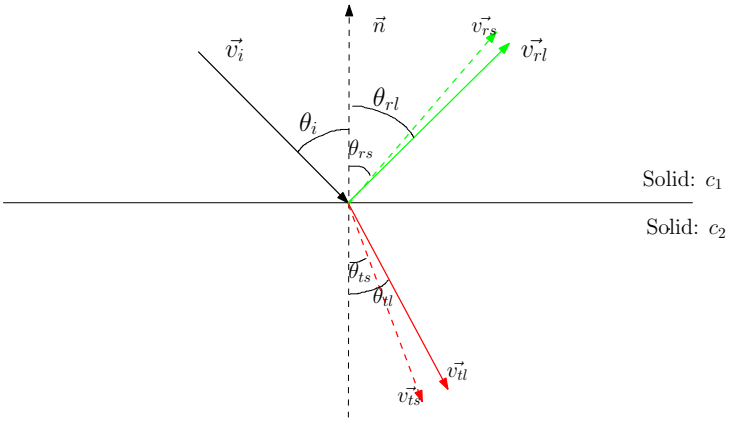


Figure 2.14: Reflection and transmission at a solid-solid interface

2.5 Ray Tracing Simulation

The algorithm to trace the ray in the sampling box developed in this study is a variant of Bresenham line drawing algorithm [3] but for 3D geometry. It is implemented as a class method of `pyHIFU.box.Box`. The algorithm can be divided into two phases: initialization and incremental traversal. To make it more intuitive, the algorithm is illustrated in 2-dimension first. Consider Figure 2.15, the algorithm should visit the voxels in the same order as their labels [1].

The initialization phase finds the coordinates and voxels where the intersection with the box starts and ends, in Figure 2.15 it is voxel 1 and voxel 19 respectively [14]. If the ray doesn't intersect the box, the ray is not processed. If the ray intersects the box, there are two situation to consider: whether the ray starts or ends inside the box or not. The routine (Appendix B) for

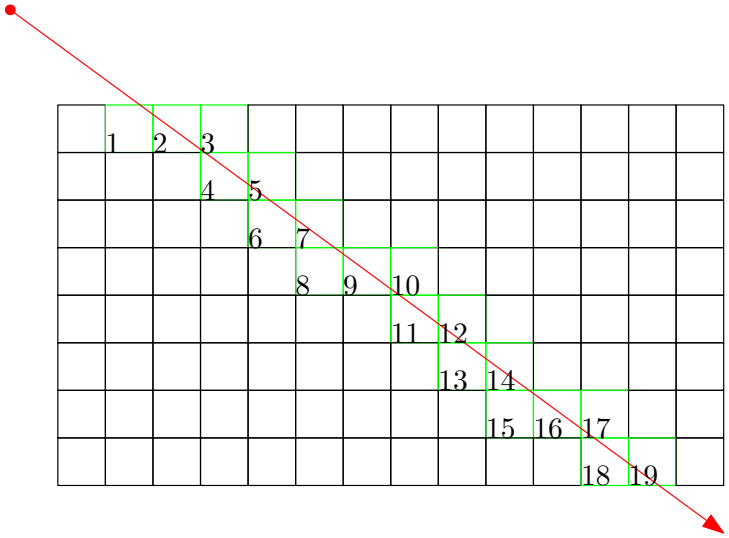


Figure 2.15: A ray intersects a 2D raster, green square indicating the interaction.

calculating the intersection of the ray with the box returns the distances from the two intersection points to the origin of the ray, in this report the distance from the start of intersection is referred to as st, the distance from the end of the intersection as et. For example, st and et are the distance from the origin of the ray to the first and last horizontal lines in Figure 2.15, respectively. From the formula of the ray and st and et, the coordinate of the starting cube `[X, Y, Z]` and the coordinate of the ending cube `[X2, Y2, Z2]` can be determined (Appendix A). Variables `StepX, StepY, StepZ` are initialized to 1 if their direction is incremental and -1 otherwise. Next, we compupate variables `tDeltaX, tDeltaY, tDeltaZ`. `tDeltaX` is the distance to move along the ray between two adjacent $y-z$ planes, similarly for `tDeltaY` and `tDeltaZ`. Finally we compute the distance to move along the ray from the origin of the ray to the next $y-z$ plane as `tMaxX`, again similarly for `tMaxY` and `tMaxZ`.

The incremental phase is similar to other digital differential analyzer algorithms. It keeps track of the variables `tMaxX, tMaxY, tMaxZ` and updates them with every increment of `[X, Y, Z]`, the coordinate of the axis-aligned bounding boxes system within the sampling box. `tMaxX, tMaxY, tMaxZ` indicates the plane where the closest intersection point is located. For example, if `tMaxX` is the smallest among the three, the ray should intersect the next cube on the x direction. First, increment to `[X, Y, Z]` is performed according to `tMaxX, tMaxY, tMaxZ`. After the increment on coordinate, the corresponding distance (`tMaxX, tMaxY, tMaxZ`) should be updated to the distance of the next plane in its corresponding direction. For the x direction, `tMaxX` should be incremented by `tDeltaX`. Algorithm 2 demonstrates the incremental phase in 3D geometry. This algorithm is a strictly adjacency ray tracing algorithm. The boxes have to have common face to be considered adjacency. The algorithm always favours the box closer to the ray direction if the ray intersects right at the box edge.

Due to the precision of float number, the coordinates of the intersection points calculated from the algorithm does not always mathematically perfectly sit on the cubes. A routine to handle

this problem is addressed in Appendix A.

Algorithm 2: Incremental phase of the ray tracing algorithm

```
1  while not reached the end [X2, Y2, Z2] do
2      process [X, Y, Z] using the function handle
3      if tMaxX < tMaxY then
4          if tMaxX < tMaxZ then
5              tMaxX = tMaxX + tDeltaX
6              X = X + stepX
7          else
8              tMaxZ = tMaxZ + tDeltaZ
9              Z = Z + stepZ
10     else
11         if tMaxY < tMaxZ then
12             tMaxY = tMaxY + tDeltaY
13             Y = Y + stepY
14         else
15             tMaxZ = tMaxZ + tDeltaZ
16             Z = Z + stepZ
```

2.6 Measurement

Measurement is conducted after the ray casting process is finished. `cast` method from transducer element class `TElement` returns a dictionary where the rays are sorted according to their `bundle_identifier`. There are two variables that are required to calculate the final result, average intensity of a bundle, average phase of a bundle. First, the rays are sampled on the sampling box in bundles. For every ray r_i in bundle B_i, the sampling routine goes as follows:

Algorithm 3: Sampling routine of one bundle B

```
1  pc = zeros(dimension of box)
   /* pressure (complex number)                                    */
2  foreach bundle B do
3      I = zeros(dimension of box)
4      ph = zeros(dimension of box)
5      counter = zeros(dimension of box)
6      foreach ray r ∈ B do
7          update I
8          update ph
9          update counter
10     foreach coordinate [x, y, z] in I do
11         pc+=sqrt((2·Z·I(x,y,z))/N_B) · e^(i·φ/N)
```

The pressure caused by one bundle B to a cube `(x, y, z)` is

$$pressure = \sum_{tr \in B} \sqrt{\frac{2 \cdot Z \cdot I(x,y,z)}{N_B}} \cdot e^{\frac{i \cdot \phi}{N}} \tag{2.13}$$

Z is the attenuation factor, N_B is the number of trident rays in bundle B.

There are two methods to update the intensity I in the first loop in Algorithm 2. Consider in the Figure 2.16, the first method is to find the perpendicular foot F of the center C of the box on ray $\vec{AB}$. The intensity at point F is used as the contribution of this ray to the box. The secound method is a weighted contribution of the intensity. First, the intensity I_F at point F is measured in the same way as in the first method. Then, the length of the intersection of the ray with the box $|AB|$ is calculated. The contribution of the ray $\vec{AB}$ to the box would be $I_F \cdot \frac{|AB|}{D}$ where D is the length of the diagonal of the box.

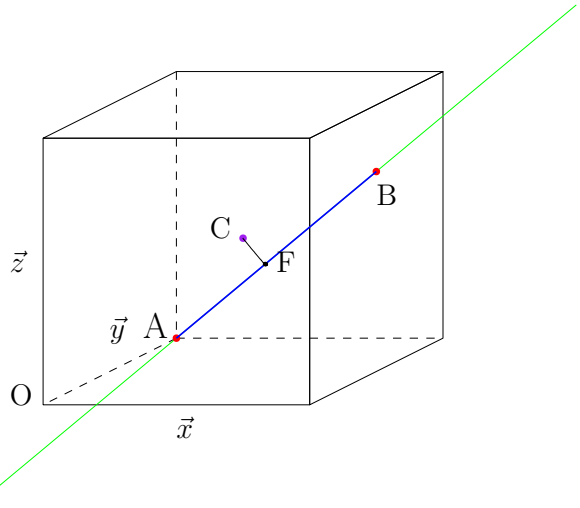


Figure 2.16: Ray intersection with a cube. A: entry point, B: exit point, C: center of the cube, O: lowest point of the cube, F: perpendicular foot on ray through C, $CF \perp AB$.

Errors are introduced to the result when using ratio of the length of the intersection segment to the length of cube diagonal as the weight. The resulted intensity is not in SI unit [10]. However, we still can compare the result with the other simulation methods (Section 3). This study only focus on the simulation of longitudinal wave in fluid media with one interface.

3 Results

In this study, there are two test cases. The first and the most simple one is that the media only consists of the Castor oil (start media in transducer). The second case is that the media consists of Castor oil and muscle. Solid media are not implement in this study yet. The resulted pressure only covers the area around the actual focus of the transducer (Figure 2.4). The initial intensity of every ray is the accurate value from the far field approximation 3.1. However, some error factors are introduced in the sampling process. Neither of the two methods to calculate contribution can produce physically realistic results. Hence, a scaling factor is required to scale the result back to the real physical unit.

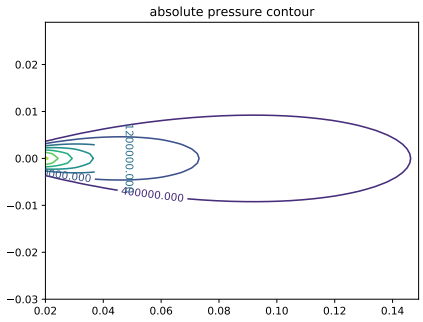


Figure 3.1: Far field approximation profile of one transducer element, start of x axis is the minimal distance at which the far field approximation holds. At the begining of every ray, the pressure is calculated from the far field approxmiation.

In this study, the result from Modena et al [11] is assumed as the gold standard. All the results from this study are compared against the gold standard. Despite the requirement of a scaling factor, however, the efforts in this study only try to prove that this method is possible for the first and the second test case. Hence, the comparison is based on similarity of the shape of the resulted pressure.

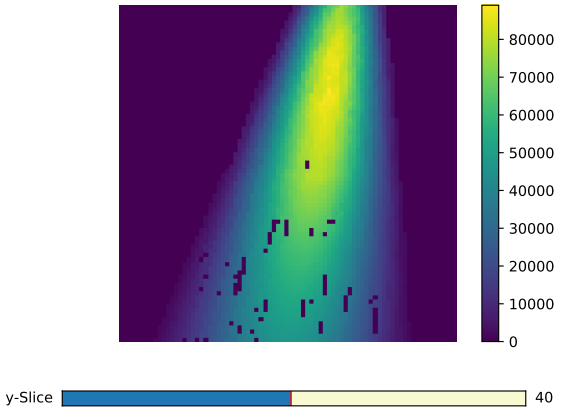


Figure 3.2: Pressure caused by one transducer. Multiple blank points appear because a small number of rays cannot cover all the area.

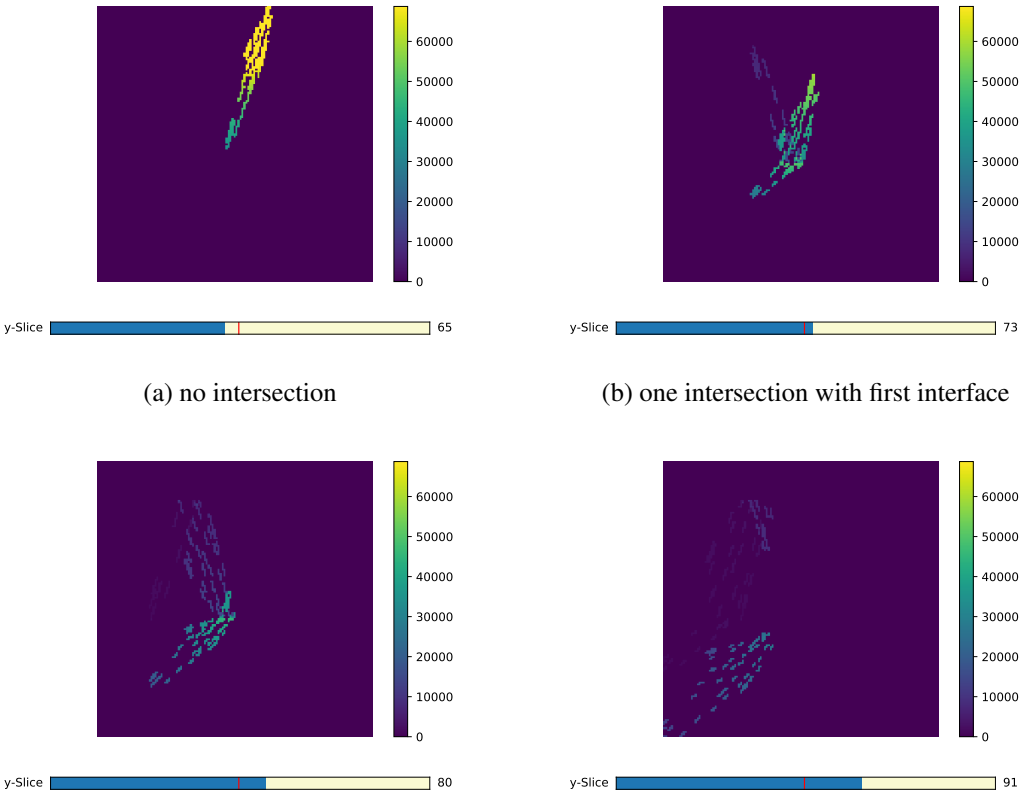
The gold standard is a matrix with the dimension of 75 by 30 by 30. The result in this study

is cropped to the same dimension to compare against gold standard. The similarity metric employed in the study is the P-norm distance, in this case $p = 2$.

$$D = \left(\sum_{i=0}^{n} |a_i - b_i|^p \right)^{1/p} \tag{3.1}$$

In equation 3.1, a_i stands for the result in this study and b_i is the gold standard. An analysis of variance is performed on the three determining parameters of the result, number of rays, trident angle and theta (Section 2.2).

The pressure is visualized as a heat map. First, we visualize the process of transmission and reflection of the ray originating from one transducer only (Figure 3.3). The color bar shows the pressure at the position. Figure 3.3a show the state when the rays just emitted from the transducer element. It can be observed that the pressure suddenly decreases after passing through the interface. The rays reflected back into the Castor oils are discarded. Figure 3.3b shows the state when the ray hits the second interface. It produces a nice illustration of reflection and refraction due to the large difference in speed of sound in the muscle and the bone. Figure 3.3c and figure 3.3d display the continued path of the reflected ray and refracted ray and their intersection with interfaces, respectively. When there's only one transducer element sending out rays, the pressure is always higher near the transducer. However, if all the transducer are activated, the superposition of all the transducer near the focus area will produce higher pressure.



(a) no intersection

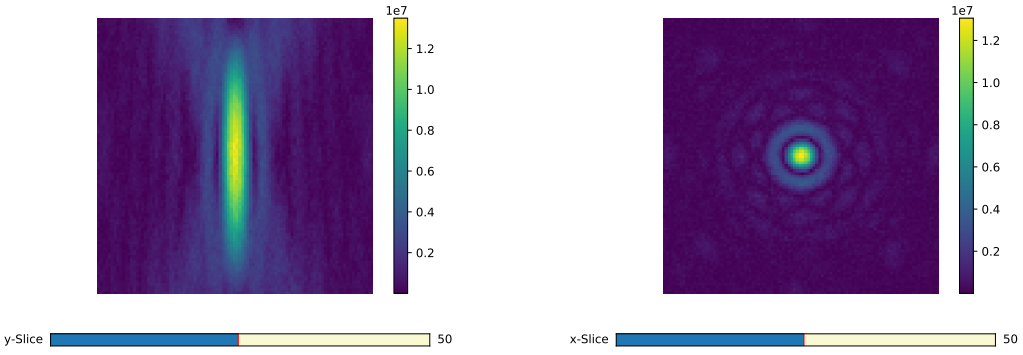
(b) one intersection with first interface

(c) continuation of new rays

(d) second intersection

Figure 3.3: Four moment during reflection and refraction. The slice number increases with the direction of the path of the trident.

An advantage feature of ray tracing model is that it takes the interference of the wave into consideration. Wave interference is the superposition of two or more wave to form a wave of greater, lower or the same amplitude. In this study, the support of interference lies in the implementation of phase and pressure decay. The interference pattern proves the correctness of the model. As can be observed from Figure 3.4a, the interference pattern is symmetric along the center axis. The interference pattern in Figure 3.4b is centrosymmetric. Both of the two figures is consistent with the organization of the element on the transducer.



(a) Slice along y axis

(b) Slice along x axis

Figure 3.4: The focus area in case 1 (two media and one interface). The interference patterns can be observed.

An experiment is set up to include three media in the geometry, Castor oil, muscle and bone. The actual focus area falls in the muscle, close to the muscle-bone interface. As a result, the pressure will include not only the interference between the rays cast from the transducer, but also the interference between incident rays and reflected rays at the focus areas. In Figure 3.5, the pressure at the focus area in this situation is displayed, in $-z$ direction in Figure 2.4. Below the sampling window is the muscle-bone interface. Above the sampling window is the transducer. The interference between the incident rays and the reflected rays result in horizontal interference pattern, symmetric against a central plane. This can also prove the correctness of the ray casting model.

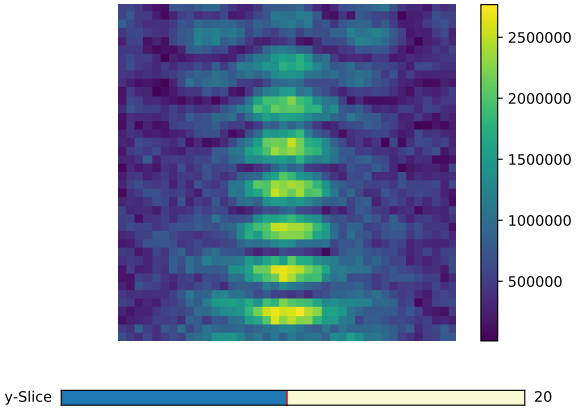
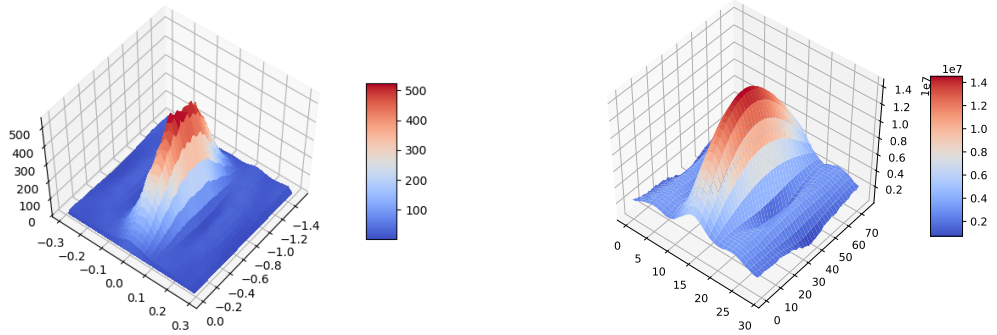


Figure 3.5: Pressure with more complex interference.

The pressure can also be visualized as surface, which is also the method employed by Modena et al [11]. Figure 3.6 show the surface plot of one slice of the pressure data sampled from the geometry (same data as Figure 3.4a). Although the unit scale is different for the two surfaces, we can compare them based on how similar are these two surfaces. In Figure 3.6, the number of rays in both (a) and (b) are 20000. It can be observed that the method presented in this study produces a smoother result than the method by Modena et al [11].



(a) Result of Modena et al.

(b) Result in this study.

Figure 3.6: The focus area in case 1 (two media and one interface). The interference patterns can be observed.

To compare the similarity the result in this study with gold standard, three analysis of variance studies are conducted. There are many parameters which can affect the result. The most important three are number of rays n_rays, theta max θ_{max} and trident angle. From previous study, it seems that more rays can yield better result. However, it also increases the computational time. To find an optimum number of rays which not only produces a result similar to the gold standard but also saves computational time, the number of rays is sampled in the range from 1000 to 20000 at the step of 500.

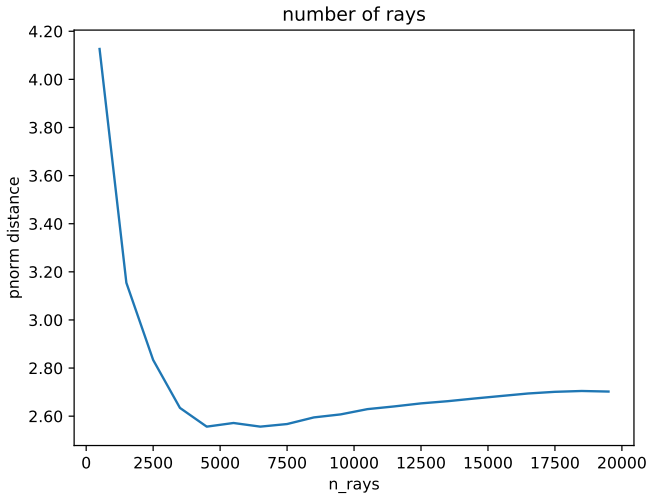


Figure 3.7: Analysis of parameter n_rays.

The similarity metric is calculated for each number of rays (Figure 3.7). The 3-norm distance is minimal at around $n\_rays = 4500$. The less the distance is, the more similar the result in

this study is to the gold standard. Before $n\_rays = 4500$, the distance drastically decreases. After $n\_rays = 4500$, the distance increases at a slower pace. This phenomenon can be interpreted as that the trident ray method produces the result most similar to the gold standard at $n\_rays = 4500$. When n_rays is more than 4500, the trident ray method can produce an even smoother result than the gold standard (Figure 3.6). In the following experiments, the number of rays is 4500.

Theta max θ_{max} determines the spreading angle of the rays from one transducer element. When the number of rays is determined, smaller spreading angle can cast more rays to the focus area. However, θ_{max} is also required to be large enough to make sure that every cube in the sampling box is covered. A similar study as before is conducted for parameter θ_{max}. θ_{max} is incremented from 0.01 to 0.1, 0.01 at each step (Figure 3.8). Similarly, the distance plot is very similar to that of n_rays. The minimal distance is achieved at $\theta_{max} = 0.05$. From $\theta_{max} = 0.01$ till $\theta_{max} = 0.03$, the distance decreases faster. This is due to that the θ_{max} is too small to cover the entire sampling space. These experiments only analyze the separate behaviour of different parameters. Although $\theta_{max} = 0.05$ might not be the optimal value, it is used in the following experiments.

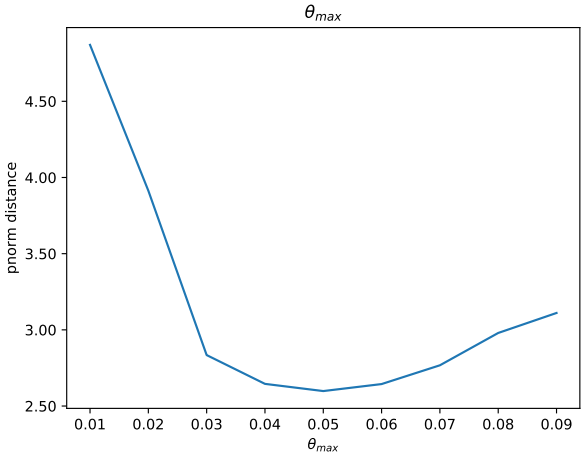


Figure 3.8: Analysis of parameter θ_{max}.

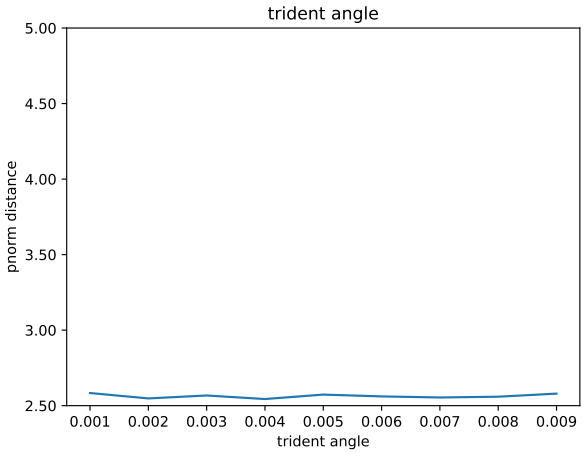


Figure 3.9: Analysis of parameter trident angle.

The third parameter to be analysed is the angle between the power ray and the auxiliary ray within a trident. Figure 3.9 shows the result of changing the trident angle on the pnorm distance. However, it doesn't show any evidence of an optimal value. The experiment in this study uses 0.005 as the optimal trident angle.

4 Discussion

Different from the previous studies, the determining factor of the pressure in this study is how many different bundles of rays pass through a cube, instead of the number of intersected rays. A bundle is determined by origin transducer element, the history of the media and type of the rays (longitudinal or shear). There are many rays in each bundle, but not as many if they are required to pass through the same sampling cube. Although increasing the number of rays still tends to produce a better simulation quality, the relation is no longer simple.

When running in serial, the model is about 2.5 times faster than the old method with the same number of rays. when the ray number is 200, the result is already showing a cigar shaped focus area. The code in this model is also optimized for parallelization. Each transducer can be allocated on one separate process. The parallelization is implemented in python using its `multiprocessing` library. This study is conducted on a workstation with Intel® Xeon® CPU E3-1230 v5 3.40GHz. The CPU has 4 cores in total (8 cores if considering virtual CPU), hence, the simulation creates a process pool with 8 workers. In this way, the simulation is approximately 4 time faster than the simulation run in serial. The reason why it is not 8 times faster is probably the data exchanges between processes cost a longer time. The program can be further optimized if the ray-box intersection and sampling is implement on GPU, as the algorithm is almost embarrassingly parallel in that regard.

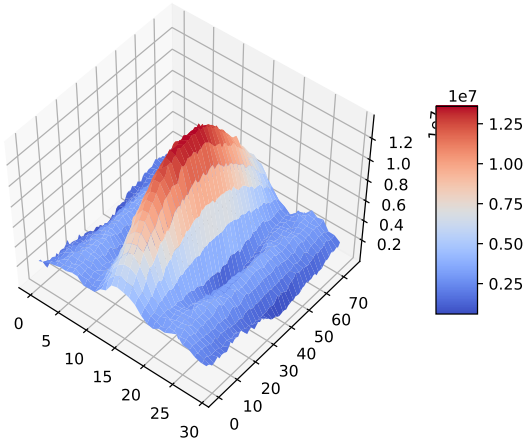


Figure 4.1: Result from this study that is closest to gold standard (4500 rays per transducer element, simulation time 2hrs)

This study introduces a new way to simulate the propagation of ultrasound ways in HIFU system using beam spreading. The goal of this study was to prove the feasibility of this new method. The model is based on acoustics laws and is capable of handling the interference. The model is tested on a simple geometry. It could be extended to support more complicated geometries. In application, the geometry model usually consists of triangular mesh with plane boundaries, which are compatible with the plane implemented in this study. However, the porous structure of the bone might be a challenge to the model [13] because the reflection and refraction could be more complicated than that in this study. Further research is needed to determine the scaling factor to scale the result in this study to physical value. The pressure can then be converted to temperature in the geometry.

A Float Number Precision

When calculating the coordinates of the intersection points, due to the limit of float number precision, some errors might occur. In this case, a positive constant ϵ is introduced into the model. If a number is less than ε or greater than $-\varepsilon$, it is regarded to be 0. For example, to prove that a point p_1 is on a plane P, one needs to the line formed by the point p_1 and an arbitrary point on the plane p_2 is perpendicular to the normal vector n of plane P. When defining the plane, a member variable named p is assigned, in this case, p is chosen as p_2. For various reasons such as that the normal vector of the line L is normalized to length 1, the dot product of n and vector $p_1\vec{p}_2$ is not 0 but a number very close to 0. In other words, even if the point should be on the plane if calculated by hand, the answer from the computer program might not be so.

Since all of the precision problems occur in vector calculation, a special class `Vec3` is implemented with ε in consideration. Experiments shows that the $\varepsilon = 1^{-10}$ is an approperiate value for the HIFU simulation.

```
def are_equal(a, b):
    # if the vectors a and b are equal, return True
    return all(np.abs(a - b) < EPS)

def are_parallel(a, b):
    # if the vectors a and b are parallel, return True
    reture all(np.cross(a, b) < EPS)
```

Another problem arises in the sampling phase. Given the coordinates of the intersection between the power ray of a trident ray and a sampling box, one should be able to find the index of the entry box and the exit point. However, because of the problem of float number precision, simply flooring the ratio of the offset to the length of cube edge might not always give the correct result. For example, in Figure A.1, there are two cases.

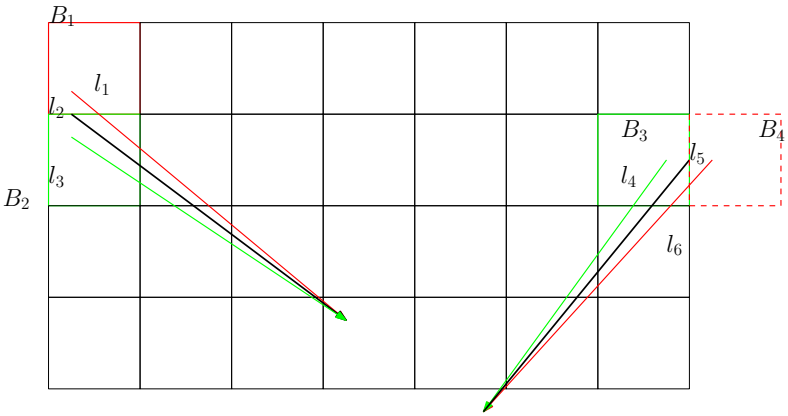


Figure A.1: Two examples of intersection with sampling box, the black lines, the green lines and the red lines are the line represented in the computer memory with errors.

First, suppose l_2 is the actual line lying exactly on the cube face. It is represented in the computer memory as l_1, whose starting point is in B_1, with the distance to the lower face of B_1 being ε. In this case, the correct entry cube is B_2. However, the ratio of the offset to the edge length is $\frac{1-\varepsilon}{1}$, which will be floored to 0. This gives a wrong entry box B_1. Second, real line l_2 is represented in computer memory as l_3, whose starting point is ε away from the upper face of B_2. Flooring $\frac{1+\varepsilon}{1}$ would give the corret result. Now let's take a look at the right part of

Figure A.1. Similarly, l_5 is the actual line mathematically. If it is represented as l_4 in computer memory, the result would be correct. If it is represented as l_6, the entry point would be B_4. Of course, if the actual ray is enough far away from the cube faces, there's no need to consider this float precision problem.

This problem can be solve by rounding the ratio according to the direction of the ray. Each case of green representation and red representation would be handled separately.

Algorithm 4: Routine of rounding x dimension of entry point.

Input : a: x coordinate of entry point, b: length of x dimension of cube, s: symbol of direction on x dimension

Output: q, r

```
r = a % b
q = a // b
if 0 < r < ε then
    if s<0 then
        q ← q − 1
        r ← b
    else
        r ← 0
else if b − r < ε then
    if s>0 then
        q ← q + 1
        r ← b
    else
        r ← 0
```

The method to round the end point is just the opposite of that of the entry point.

B Ray Box Intersection

Algorithm 5 is implemented as a member function of the class `Box`. This function is called for every ray in the model thus needs optimizing. The algorithm is only suitable for axis-aligned bounding boxes.

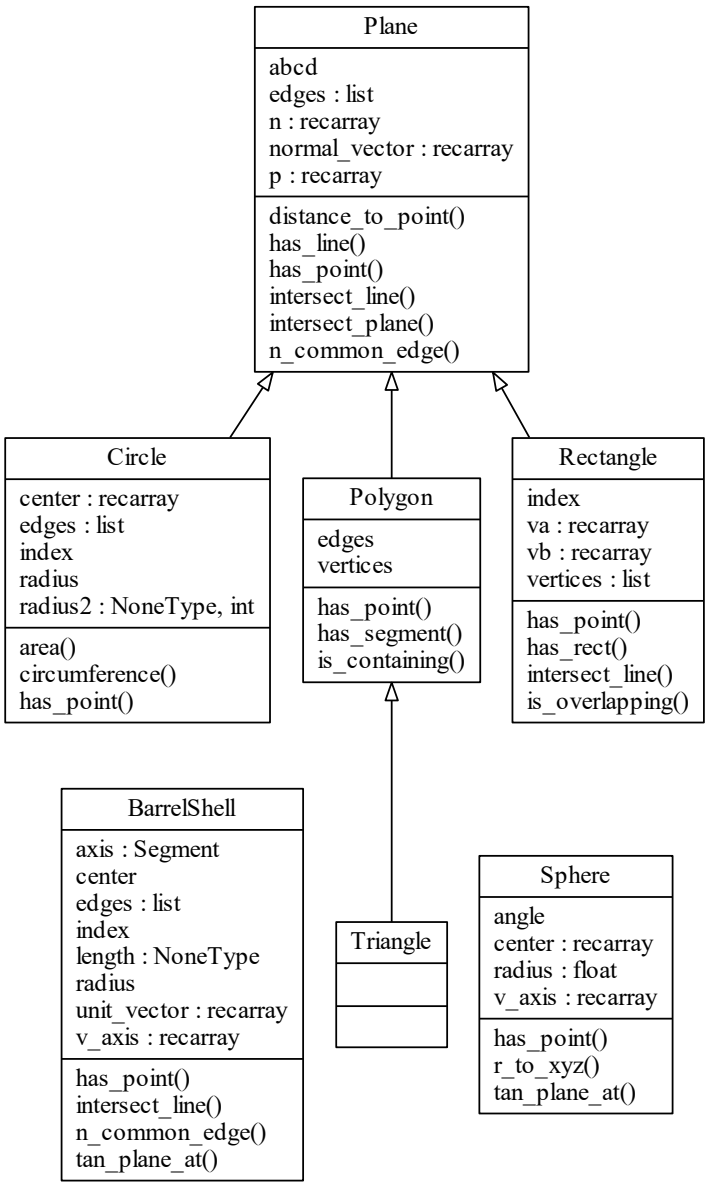
Algorithm 5: Ray intersects with an axis-aligned bounding box.

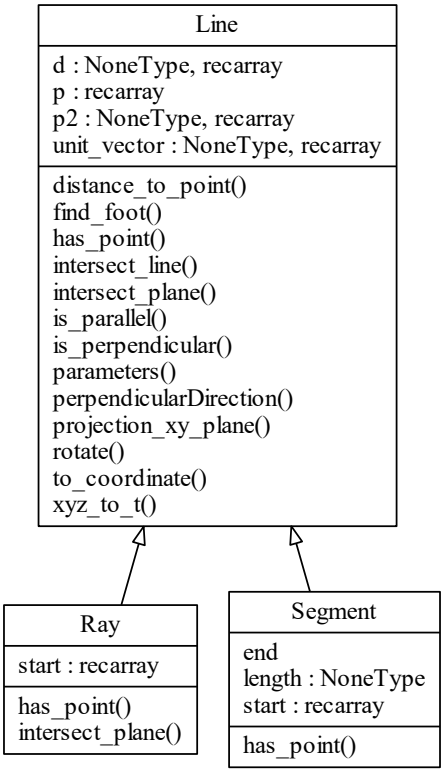
Input : ray, box
Output: two intersection points

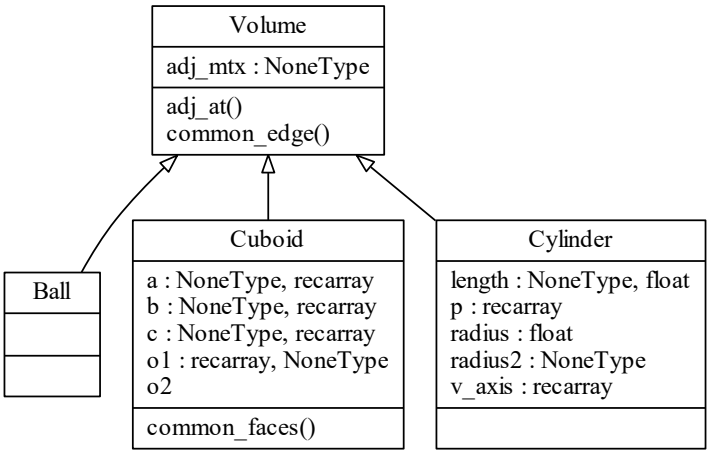
```
1  tx1 = (box.min.x - ray.x0.x)/ray.n.x
2  tx2 = (box.max.x - ray.x0.x)/ray.n.x
3  tmin = min(tx1, tx2)
4  tmax = max(tx1, tx2)
5  ty1 = (box.min.x - ray.x0.x)/ray.n.x
6  ty2 = (box.max.x - ray.x0.x)/ray.n.x
7  tmin = max(tmin, min(ty1, ty2))
8  tmax = min(tmax, max(ty1, ty2))
9  tz1 = (box.min.x - ray.x0.x)/ray.n.x
10 tz2 = (box.max.x - ray.x0.x)/ray.n.x
11 tmin = max(tmin, min(tz1, tz2))
12 tmax = min(tmax, max(tz1, tz2))
13 if tmax > tmin then
14 |   return tmax, tmin
15 else
16 |   return None, None
```

C Geometry

UML graphs of the geometric classes. The blank classes are not implemented yet.




Line
d : NoneType, recarray
p : recarray
p2 : NoneType, recarray
unit_vector : NoneType, recarray
distance_to_point()
find_foot()
has_point()
intersect_line()
intersect_plane()
is_parallel()
is_perpendicular()
parameters()
perpendicularDirection()
projection_xy_plane()
rotate()
to_coordinate()
xyz_to_t()
Ray
start : recarray
has_point()
intersect_plane()
Segment
end
length : NoneType
start : recarray
has_point()


Volume
adj_mtx : NoneType
adj_at()
common_edge()
Ball
Cuboid
a : NoneType, recarray
b : NoneType, recarray
c : NoneType, recarray
o1 : recarray, NoneType
o2
common_faces()
Cylinder
length : NoneType, float
p : recarray
radius : float
radius2 : NoneType
v_axis : recarray

References

[1] John Amanatides and Andrew Woo. "A Fast Voxel Traversal Algorithm for Ray Tracing". In: *Proceedings of EuroGraphics* 87 (Aug. 1987).

[2] B.A. Auld. *Acoustic fields and waves in solids*. A Wiley-Interscience publication. Wiley, 1973. ISBN: 9785885013437.

[3] J. BRESENHAM. "A linear algorithm for incremental digital display of circular arcs". In: *Communications of the ACM* 20.2 (1977), pp. 100–106. URL: https://ci.nii.ac.jp/naid/80013662045/en/.

[4] H M M ten Eikelder et al. "Modelling the temperature evolution of bone under high intensity focused ultrasound". In: *Physics in Medicine and Biology* 61.4 (Feb. 2016), pp. 1810–1828. DOI: 10.1088/0031-9155/61/4/1810. URL: https://doi.org/10.1088%2F0031-9155%2F61%2F4%2F1810.

[5] *High-Intensity Focused Ultrasound on Wikipedia*. https://en.wikipedia.org/wiki/High-intensity_focused_ultrasound.

[6] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing In Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[7] Jürgen W. Jenne, Tobias Preusser, and Matthias Günther. "High-intensity focused ultrasound: Principles, therapy guidance, simulations and applications". In: *Zeitschrift für Medizinische Physik* 22.4 (2012). Schwerpunkt: Multimodale Bildgebung und Therapie, pp. 311–322. ISSN: 0939-3889. DOI: https://doi.org/10.1016/j.zemedi.2012.07.001. URL: http://www.sciencedirect.com/science/article/pii/S0939388912000943.

[8] Eric Jones, Travis Oliphant, Pearu Peterson, et al. *SciPy: Open source scientific tools for Python*. 2001–. URL: http://www.scipy.org/.

[9] Julius Koskela et al. "Stochastic ray tracing for simulation of high intensity focal ultrasound therapy". In: *The Journal of the Acoustical Society of America* 136.3 (2014), pp. 1430–1440. DOI: 10.1121/1.4892768. eprint: https://doi.org/10.1121/1.4892768. URL: https://doi.org/10.1121/1.4892768.

[10] W. Kreider et al. "Characterization of a multi-element clinical HIFU system using acoustic holography and nonlinear modeling". In: *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control* 60.8 (Aug. 2013), pp. 1683–1698. ISSN: 0885-3010. DOI: 10.1109/TUFFC.2013.2750.

[11] D Modena et al. "Modeling the interference between shear and longitudinal waves under high intensity focused ultrasound propagation in bone". In: *Physics in Medicine & Biology* 63.23 (Dec. 2018), p. 235024. DOI: 10.1088/1361-6560/aaef14. URL: https://doi.org/10.1088%2F1361-6560%2Faaef14.

[12] Pascal Ramaekers. "Methods for abdominal tumor ablation through image-guided focused ultrasound". PhD thesis. Utrecht University, 2017.

[13] J.H. van Wijk. "Validation and expansion of a model to simulate the effect of ultrasound on bone and tissue". In: (Sept. 2013).

[14] Amy Williams et al. "An Efficient and Robust Ray-Box Intersection Algorithm". In: *J. Graphics Tools* 10 (Jan. 2005), pp. 49–54. DOI: 10.1145/1198555.1198748.