Conjectured

Can gravitational waves be longitudinal and slow?

ir S.J. van Stekelenburg

21 February 2019

Abstract

Experiments have been done on the behaviour of an array of numbers, representing the behaviour of an elastic fabric in time. Current literature on gravitational waves considers only transverse gravitational waves due to relativistic arguments. The array software simulates a complete machinery. Current models lack complete machinery and resort to mechanisms. For example through some undefined mechanism energy is distorting spacetime and through some mechanism this spacetime interacts with the energy to change its motion. Describing spacetime behaviour by means of an equation does not provide a complete machinery. In this experiment it is shown that if the machinery of just a 3 dimensional elastic space is assumed (for a more detailed description see [1] and [2]) one should seriously reconsider the negation of longitudinal gravitational waves. The numeric fabric behaves in such a manner that longitudinal and transverse waves move with the same speed through space. This is fundamentally different from descriptions in literature of waves in homogeneous isotropic media where longitudinal waves travel faster by a factor of $\sqrt{2}$. The difference seems to be in the assumptions that Poissons's ratio is always greater than zero ([3]) and that the divergence of space volume has to be zero due to special relativity. For the numeric elastic fabric under investigation the Poisson ratio is zero. It is also not possible to have zero divergence of volume because change is only expressed as change in volume through stretch of the fabric. Special relativity does not apply because there is only space (not relative to anything else). As an aside it is suggested that the definition of a gravitational wave is broadened to also contain waves in the fabric that travel at speeds lower than the maximum speed at which waves naturally can travel through the fabric. With this adaptation the machinery is suited as a candidate for explaining the phenomena that are currently often explained by a mysterious form of energy called dark matter.

1 Introduction

Continuous elastic space has been modeled in the computer by a discrete array of points simulating a discrete elastic fabric. In an appendix some of the technical details are described. As a result of simulating the merging of the event horizons of two merging black holes by means of an ellipsoid of points that behaves (expands and contracts) as a quadrupole the question arose why current physics demands gravitational waves to be transverse.

The quadrupole produces longitudinal waves and superpositions of longitudinal en transverse waves depending on the direction relative to the axes of the ellipsoid. The fabric that has been tested is a grid of 160 (z-axis) by 52 by 52 points (x- and y-axes), representing a three dimensional torus by connecting opposing sides of the box (3-orthotope or hyperrectangle). This relatively arbitrary choice of points is mainly guided by computer memory considerations. At point (26, 26, 10) a source of energy is introduced by moving that point to another position, introducing stretch in an otherwise completely "flat" space. The energy spreads out ball-symmetrically outward from the source with a speed that is depending on the stiffness of the fabric ($speed_{wave}$ is proportional to $\sqrt{\frac{1}{stiffness}}$ as expected. $stiffness$ or elastic modulus is "this.stijfheidsParam" in a code snippet from Figure 4). Just to be clear $speed_{wave}$ is lower than the maximal possible speed of propagation of information in the discrete array (which is one unit of distance divided by one unit of time).

2 Experimental results

Three different excitations of the source were introduced. One was purely transverse (28, 26, 10), one was purely longitudinal (26, 26, 12) and one was a combination of the two (28, 26, 11). In the combination experiment both signals arrived at the same time. For some unknown reason these signals are almost the same but slightly different at some points, suggesting that there is a difference in the propagation of transverse and longitudinal waves. It seems that transverse waves can just hold their amplitude fractionally better. This calls for more research. Also the beginning (really small amplitude) tale of the transverse wave arrives a little (6 time ticks) before the tail of a longitudinal wave but the main features and energies coincide.

Table 1: *This table shows some results (28 januari 2019) from exciting the elastic fabric at point (26, 26, 10) where the point is brought to (27, 26, 11) at time T = 2. Space is flat at time T = 1.*

Position X, stiffness 0.04f	Time steps	Position X, stiffness 0.01f	Time steps
30	35	30	72
40	57	40	120
50	82	50	169
60	106	60	219
70	131	70	269
80	155	80	319
90	179	90	368
wegens torus interferentie		100	319

Appendix

A The representation of space

The software, including the array, has been constructed using C♯, .NET and XNA. The array represents a 3 dimensional space during three consecutive time steps. Two successive moments in time are used to calculate the next moment. The idea is to form a discrete frame of points in the computer to represent a volume of continuous elastic material in reality (see figure Figure 1 on page 5). The frame (like the volume) will start moving when it is deformed and let go. The elasticity of the fabric is implemented by considering the distance between a point and its neighbouring points. The attracting force (F) between two neighbouring points is related to the distance (D) between the two points. In a formula distance and force relate as $F = -RD$. R is considered a variable (within limits a constant) called variable resistance of the fabric (In Hooke's law R is a constant called the stiffness). The minus sign indicates an attractive force between the two points. To get a grip on the elastic volume's behaviour the two initial time slices of its form will have to be tuned by the AI. Than time takes over and the next new time-slice of space volume follows from the programmed elastic behaviour and shape of the two previous slices, and so forth.

A.1 The space array

The numeric fabric is an attempt to mimic reality in the computer. The model for space consists of an array of points (See figure Figure 2 on page 6). Each point is connected to twelve neighbouring points in the grid. The connection between two points can be described by a spring with constant or variable elastic modulus. When the length of all springs is equal the grid has a Face Centred Cubic structure (See for code figure Figure 3 on page 8) . Three versions in time of this space-grid are held in memory and are related as in a Markov chain of second order. This means that a new space-form at time T_3 is calculated from two

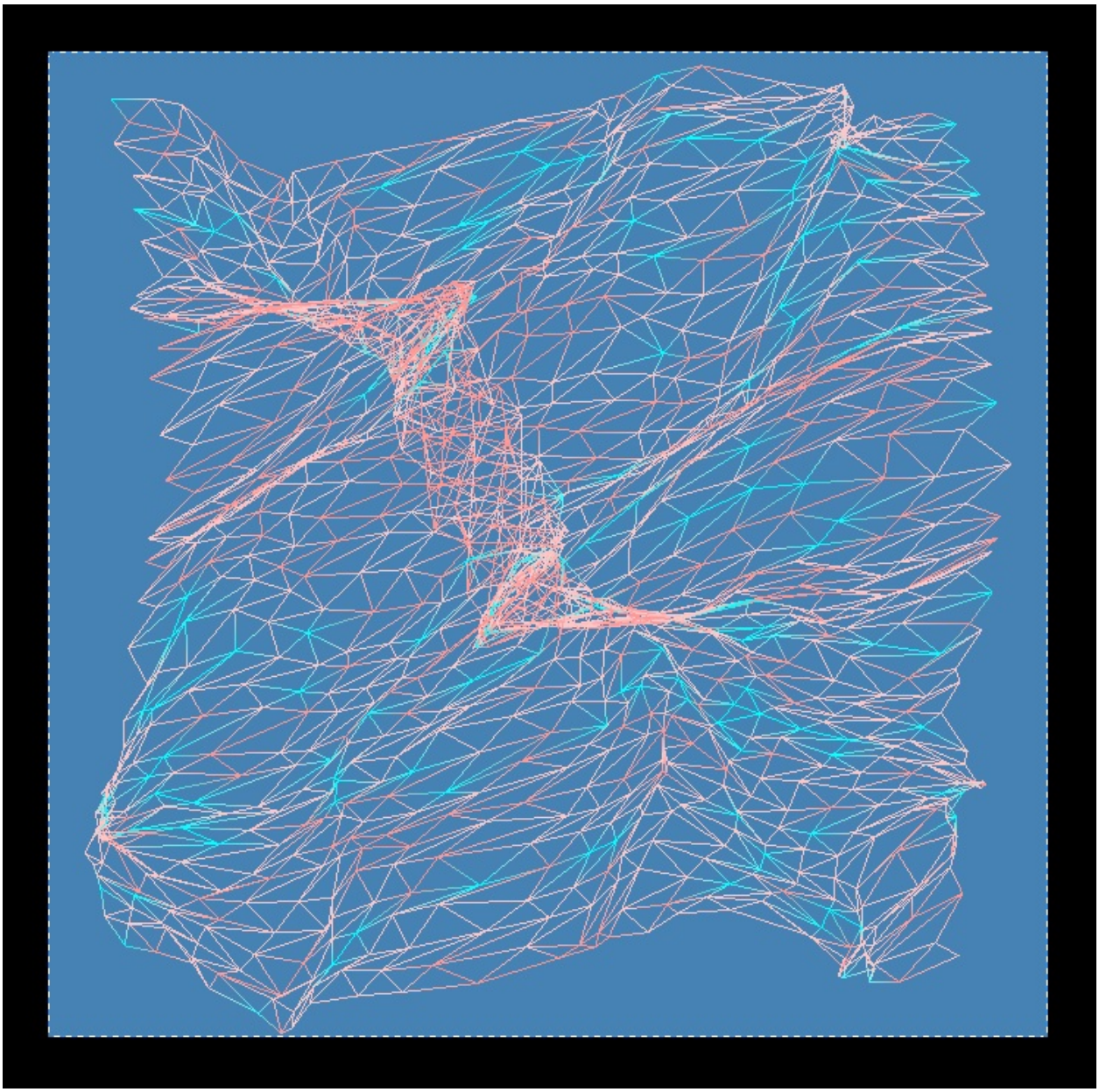

Figure 1: *A state of the array interpreted as a set of points in a frame that influence each other. As an interpretation you can see a stringy wave in a continuous 3 dimensional fabric of space. This stringy wave, defined as an object from a human perspective, moves with a speed less than the maximum speed even though the material changes are only ruled by the properties of the elastic fabric and locally everything is communicated with the same speed. Large scale waves like this stringy one are a candidate for describing phenomena that are currently described by "dark matter".*

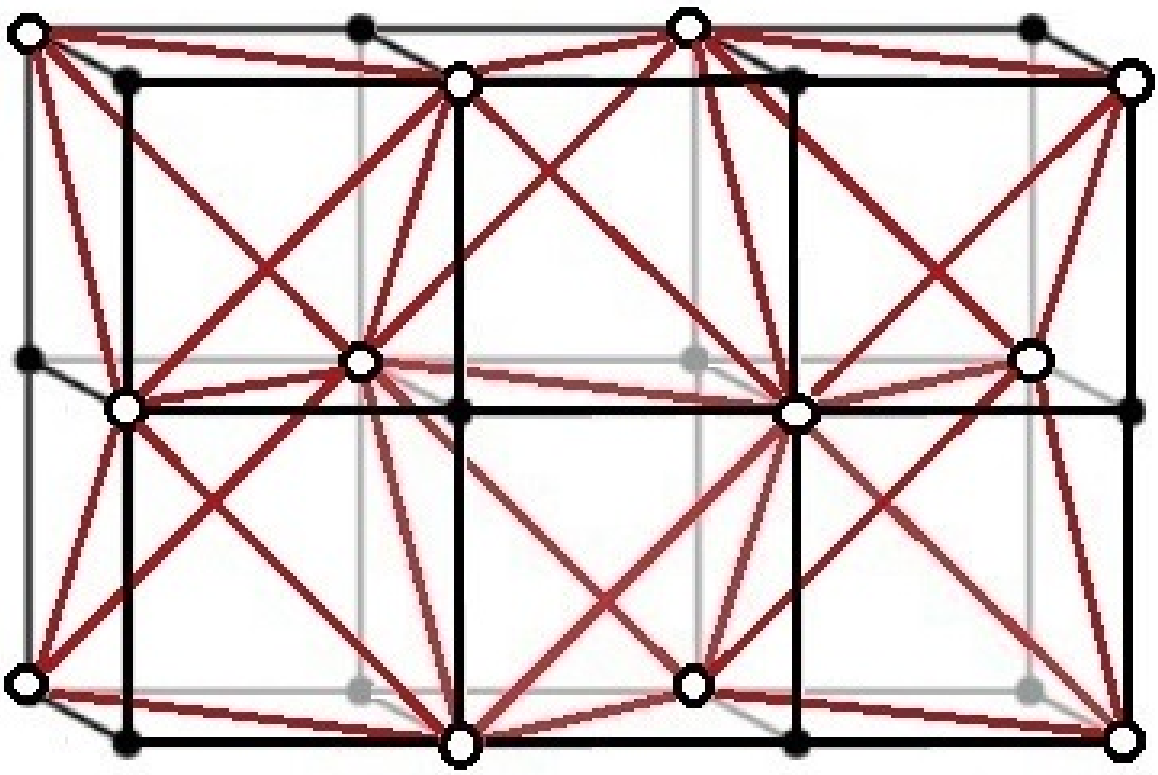

Figure 2: *A face centred cubic structure can be constructed from a cubic structure by leaving all the black points out. The red lines represent springs between neighbouring points. Each point has twelve equal neighbours so twelve red lines should end on a lattice point.*

older versions at times T_1 and T_2. The next position of a point at time T_3 is depending on its old position at time T_2, its speed at T_2 calculated from positions at times T_1 and T_2 and on the positions of the twelve neighbours at T_2. A three dimensional cube with opposite sides identified (in other words a three-torus) represents a stable structure where all points are surrounded by twelve neighbours. Alternative structures are easily realised. Balls for example, constructed from points within a certain radius (points on the surface have less than twelve neighbours), are easily constructed and will oscillate, collapse and expand, under the influence of potential and kinetic energy. Resonance cavities of all sorts of shapes, where all edges are held fixed in space and the interior is elastic space, are other examples. These face centred cubic structures (or also structures where points are interacting with 4, 6 or 8 neighbours because these structures all behave pretty much the same dynamically as is to be expected) and Markov chains in the computer are known structures. New will be the way an Artificial Intelligence will be used to manage and control these elastic numeric fabrics and their computational abilities (where no man has yet succeeded). New is the search in these fabrics for carrier topologies and particle-look-alikes with behaviour that can be explained as bosonic and fermionic behaviour.

A.2 The ‘elastic‘ behaviour of the array

The array behaves according to an algorithm that simulates elasticity. The idea is for every digital point in space to be influenced by its nearest neighbours. Each point attracts its neighbouring points according to Hooke’s law where distance and force between two neighbouring points are related. Conceptually the constant in Hooke’s law doesn’t have to be a constant (this variability of the resistance of the fabric is conjectured meaningful). The elasticity-code is shown in figure Figure 4 on page 9. It seems wise to look for other situations in which elasticity was modelled in computers. Take for instance the elastic behaviour of crystal lattices. Algorithms from other disciplines were not yet object of comparison or research.

A.3 General functionality of the existing software

The array and its behaviour have been visualised to be able to analyse and manipulate it. The 3 dimensional dynamical and graphical nature of the problem has led to the use of C, XNA and the .NET environment, an environment that facilitates both dynamics and graphics. Other more up to date software may be better suited. Space is visualised as a frame of lines or triangles, as vector-fields or as a collection of dots. Several demo examples of pseudo-physics experiments are available. With the use of a mouse and keyboard some manipulations (a camera view) such as rotations and translations of the framework are possible to investigate its form. Measures of potential and kinetic energy are used to get more feel for the nature of the behaviour of the frame. Single time step calculations, on going calculations as well as time reversal are possible. The torus (where all sides of the elastic cube are connected with their opposite sides) can be transformed into a single ’loose hanging’ cube or ball that is moving and oscillating under the influence of its internal elasticity and added potential and kinetic energy.

```
public float[, , ,] FormsArray;


protected int numberOfPointsXdirection;
protected int numberOfPointsYdirection;
protected int numberOfPointsZdirection;



public void InitialiseSpace()
{
 //This initialisation is for when each point in the grid has equal distance
 //to all its 12 neighbouring points.
 //The initialisation is the same as for a cubic grid except in this case
 //only half of the points are relevant. point 000 is in contact
 //with 200 en 020 en 002. The points in between are left out!
 //these left out points comply to (x + y +z)%2 == 1

 for (int x = 0; x < this.numberOfPointsXdirection; x++) {
  for (int y = 0; y < this.numberOfPointsYdirection; y++) {
   for (int z = 0; z < this.numberOfPointsZdirection; z++) {
    if ((x + y + z) % 2 == 1) continue;

    FormsArray[x, y, z, 0] = x;
    FormsArray[x, y, z, 1] = y;
    FormsArray[x, y, z, 2] = z;
    FormsArray[x, y, z, 3] = x;
    FormsArray[x, y, z, 4] = y;
    FormsArray[x, y, z, 5] = z;
    FormsArray[x, y, z, 6] = x;
    FormsArray[x, y, z, 7] = y;
    FormsArray[x, y, z, 8] = z;
 }}}}
```

Figure 3: *Here is the code for initialising the computerarray representing a face centred cubic structure.*

```
 public override void Calculate3Torus(int MostRecentForm)
 { int x, y, z; int teller = 0;

  if (this.mostRecentForm == 1) { XT = 3; YT = 4; ZT = 5; XTmin1 = 0; YTmin1 =
      1; ZTmin1 = 2; XTmin2 = 6; YTmin2 = 7; ZTmin2 = 8; }
  else if (this.mostRecentForm == 2) { XT = 6; YT = 7; ZT = 8; XTmin1 = 3;
     YTmin1 = 4; ZTmin1 = 5; XTmin2 = 0; YTmin2 = 1; ZTmin2 = 2; }
  else if (this.mostRecentForm == 3) { XT = 0; YT = 1; ZT = 2; XTmin1 = 6;
     YTmin1 = 7; ZTmin1 = 8; XTmin2 = 3; YTmin2 = 4; ZTmin2 = 5; }

  for (x = 0; x < this.NumberOfPointsXdirection; x++) {
   for (y = 0; y < this.NumberOfPointsYdirection; y++) {
    for (z = 0; z < this.NumberOfPointsZdirection; z++) {
     teller += 1;
     this.DetermineNeighbours(x, y, z);
     FormsArray[x, y, z, XT] =
      this.VectorPotentialX(x, y, z) + //acceleration
      + this.VectorKineticX(x, y, z) //speed
      + FormsArray[x, y, z, XTmin1]; //place

     FormsArray[x, y, z, YT] =
      this.VectorPotentialY(x, y, z) + //acceleration
      + this.VectorKineticY(x, y, z) //speed
      + FormsArray[x, y, z, YTmin1]; //place

     FormsArray[x, y, z, ZT] =
      this.VectorPotentialZ(x, y, z) + //acceleration
      + this.VectorKineticZ(x, y, z) //speed
      + FormsArray[x, y, z, ZTmin1]; //place
 }}}}

protected override float VectorPotentialX(int X, int Y, int Z)
{//Here the acceleration is related to the force in the X direction that
 //drives the point back to the centre of all its 12 surrounding points
 //(via Hooke's law).
 return this.stijfheidsParam * (this.CentrePointOfGravityMeasureXTmin1 - 12 *
    FormsArray[X, Y, Z, XTmin1]);
}

protected override float VectorKineticX(int X, int Y, int Z)
{//Here the speed in the X direction is calculated at the time we start
 //calculating the next time frame that is 1 unit ahead.
 return FormsArray[X, Y, Z, XTmin1] - FormsArray[X, Y, Z, XTmin2];
}

Code snippets:
//oaX1 is the Xcoordinate of one of the 12 neighbouring points, the one below
// and behind the point under investigation. o=below, b=on top, a=behind,
// v=in front, l=left, r=right. edgeCompensation connects points on a edge
// of the cube to point(s) on the opposing side to realise a Torus.

this.oaX1 = FormsArray[x, y, z, XTmin1];
this.CentrePointOfGravityMeasureXTmin1 = this.edgeCompensationX +
   oaX1 + baX2+ovX3+bvX4+laX5+raX6+lvX7+rvX8+loX9+roX10+lbX11+rbX12;
```


Figure 4: *Build in elasticity. Comments (text behind //) in the code are hopefully sufficient.*

References

[1] The properties of space, Bas van Stekelenburg, 2014, destekel.nl

[2] Can Artificial Intelligence bring life in space, Bas van Stekelenburg, 2018, destekel.nl

[3] The Feynman lectures on physics, Richard Feynman, ISBN 0-201-02116-1