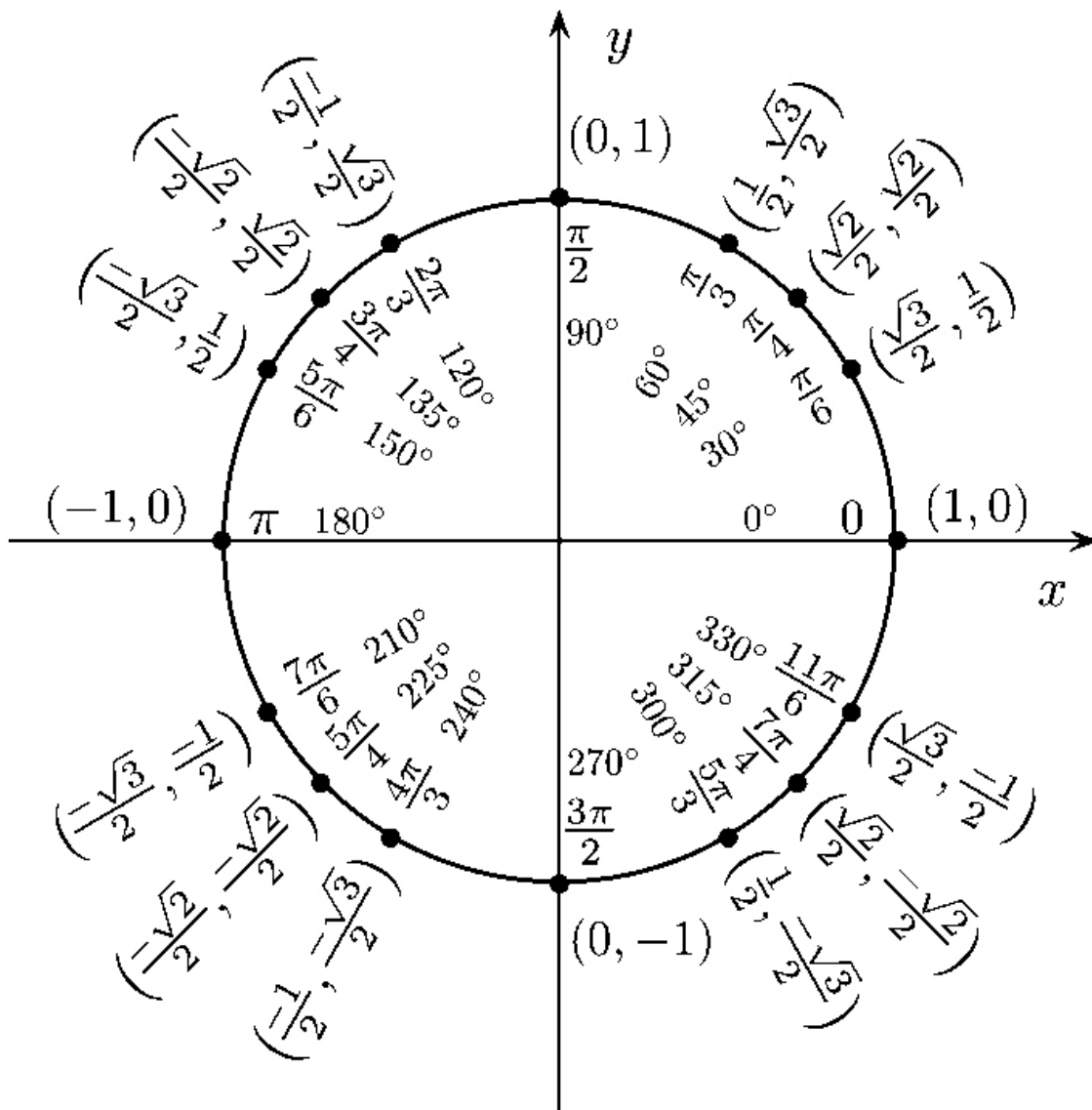


Metric Science

Gerhard Jan Smit and Jelle Ebel van der Schoot

In this article a particle is being presented that explains all known forces of nature. The particle has no dimensions, it is a dimensional basic particle. Hence it gets the following name: 'dimensional basic' (db) particle. The core of this discovery is that the separate fundamental forces of nature: - the strong interaction, the electromagnetic interaction, the weak interaction and the gravitational interaction - are calculatable with one formula out of one principle. The statistical math of the quantum theory is set aside in favor of a goniometric approach. Gravitation is the only force that matters and the strong force, the electromagnetic force and the weak force can be explained out of gravitation while gravity itself is only caused by the curvature of db's.

The formula for the extent of curvature around a db is: $\sqrt{x^2+y^2+z^2} \times Kr = 1$. In the formula: x, y, z, are coordinates in spacetime [m], Kr = curvature [m^{-1}].



Contents

| | |
|--|----|
| Outline of observed conflicts within quantum mechanics | 3 |
| Dimensional Basic | 5 |
| Coding the dimensional basic | 6 |
| The photon | 9 |
| Cosmological consequences of the photon as a two-db-system | 13 |
| Black holes | 14 |
| Electrons | 15 |
| Quarks, protons and neutrons | 16 |
| Hawking radiation | 19 |
| More complex particles | 21 |
| Positron Emission Tomography (PET) seen in a new light | 22 |
| Electromagnetic fields | 25 |
| Memorandum about quantum field theory (QFT) | 27 |
| Acknowledgement | 28 |
| Forum | 29 |
| Addendum - Computer program sources | 32 |

Introduction

Now, for the first time, a particle will be presented in this article through which all forces are explained in a satisfactory way. It concerns the so-called dimensional basic (db or λ).

This article will start with an outline of the observed conflicts within quantum mechanics. After that, the theory will be described, the dimensional basic followed by the consequences for the photon, the electron, the quarks, the protons and neutrons, the more complex particles and the nature of electromagnetic fields. The article will finish with a justification.

Quote by Einstein: "Imagination is more important than knowledge. For knowledge is limited to all we now know and understand, while imagination embraces the entire world, and all there ever will be to know and understand."

Outline of observed conflicts within quantum mechanics

It seems an impossibility to indicate the properties of a macroscopic object using quantum logic. The properties of microscopic elementary particles that are known at this time make this very difficult. Elementary particles have properties that cannot be defined, or only in a complex way. One significant problem is that the gravity at the level of the elementary particles will not be straight-jacketed into the Standard Model (Newton). In the macroscopic world, facts (position, speed and time) are true facts. In the microscopic world, one cannot often say that these are true or untrue. This begs the question: How well do we understand the world at the atomic scale?

For example, Werner Heisenberg claimed: "The subatomic world demonstrates again and again that we live in a psychedelic world that, to our common sense, is completely absurd."

According to the current models the world is made up of particles; this includes electrons, protons, and neutrons. Protons and neutrons are made up of constituent particles (quarks). Particles move under the influence of forces. Recognizable are the short distance force (weak interaction) and the long distance forces (strong, electromagnetic and gravitational interactions). There has been considerable progress in the search for a united theory of these forces. The description of all these particles and forces takes place within quantum mechanics. Quantum mechanics is not just another physical theory; it is a framework for all physical theories. Quantum mechanics describes the nature of the particles and the forces that interfere with each other from the particles.

In order to study the smallest building blocks of matter particle accelerators are used. In this method elementary particles are artificially accelerated and brought into collision with other particles, creating new particles. Through observation of their tracks (whether or not deflected into a magnetic field (only electrically charged particles)) and mutual collisions the properties of the particles can be studied. Does this provide us with a good picture of the

world or is our picture a description of the results of these multiple experiments? Do the experiments supply a good fundamental description of the entity of the particles?

One would like an interpretation of quantum mechanics that corresponds with the experience in the macroscopic world *and* that is represented by classic mechanics. However, the classic world is in part not consistent with the world of quantum mechanics. This leads to essential questions. Can the universe be represented by quantum mechanics? It seems a reasonable expectation that the atoms in the universe would obey the laws of physics. Currently this doesn't seem to be the case.

First of all, on the macro level there are observations of deviating speeds in galaxies. These speeds do not correspond with the directly observed matter and can only be explained by the presence of unknown mass called dark matter. From data of gravitational lenses there is strong evidence as to the presence of dark matter. These data suggest the presence of dark matter in clusters and around galaxies. Although this matter has never directly been observed the indirect evidence of its existence is overwhelming.

On a micro level too the questions are fundamental. For example, within quantum mechanics there is the unexplained phenomenon of entanglement. Two particles that simultaneously come into being – but are situated at a great distance from each other – each turn out to possess properties that correspond with each other. This would bring to mind a common cause in the classic sense. However, if the situation changes for one of the particles (e.g. the spin) then the situation will simultaneously change for the other particle. It seems as if from a distance an instantaneous transmission of information takes place. So this correlation between the two particles ostensibly goes beyond what is considered possible in classic physics. The fact that a particle does not choose a specific state until its observation (measuring) brought Einstein to remark: *“God does not play dice.”* It is clear that Einstein meant that there must be an underlying, understandable reason for the presumed transmission of information. However, to this day, a satisfactory explanation for this phenomenon has not been found.

There are also questions in which micro level and macro level both play a role. First there is the attraction of a photon by a gravitational field. A photon is deflected in its track by a heavy mass in space. Why does the photon obey to Einstein's ideas of curved spacetime? Traditionally the photon is considered to be massless, the reason why the underlying mechanism has not yet been fully understood. Then there is the gravitational redshift that a photon undergoes when close to an object with an enormous curvature. For example, near the event horizon of a black hole the redshift becomes extreme (almost infinite). Although both of these phenomena have been universally accepted and observed there is no full comprehension. Why does the photon undergo such a deflection and what is the mechanism of the gravitational redshift?

In this article an unconventional explanation is proposed which forms the foundation for the understanding of nuclear forces both on the micro as well as the macro scale.

Dimensional Basic

The axiom is that the most elementary particle in existence is the dimensional basic (db or λ). The λ itself has no dimensions (no length, no width and no height). The λ is found everywhere in the universe and is always moving through spacetime, where the speed of the movement of the λ , in respect to its surroundings, can have any value. The curvature of space on the location of the λ is infinite while time on the location of the λ stands still. The λ behaves like a black hole without dimensions. The λ is the building block of all that we perceive.

The formula for the extent of spacetime curvature around a λ is:

$$\sqrt{x^2 + y^2 + z^2} \times Kr = 1 \quad (0)$$

In the formula: x, y, z, are coordinates in spacetime [m], Kr = curvature [m⁻¹].

Formula (0) describes the relative lessened extent of curvature of spacetime surrounding the λ . In the formula the distance from a specific point in spacetime to the λ is always greater than zero. The smaller the distance the bigger the curvature, the bigger the distance the smaller the curvature.

Through agglomeration, or rather joint interaction, the λ -particles form phenomena that at a certain moment rise above the observational limit. The λ itself exists below the observational limit and so it cannot directly be demonstrated.

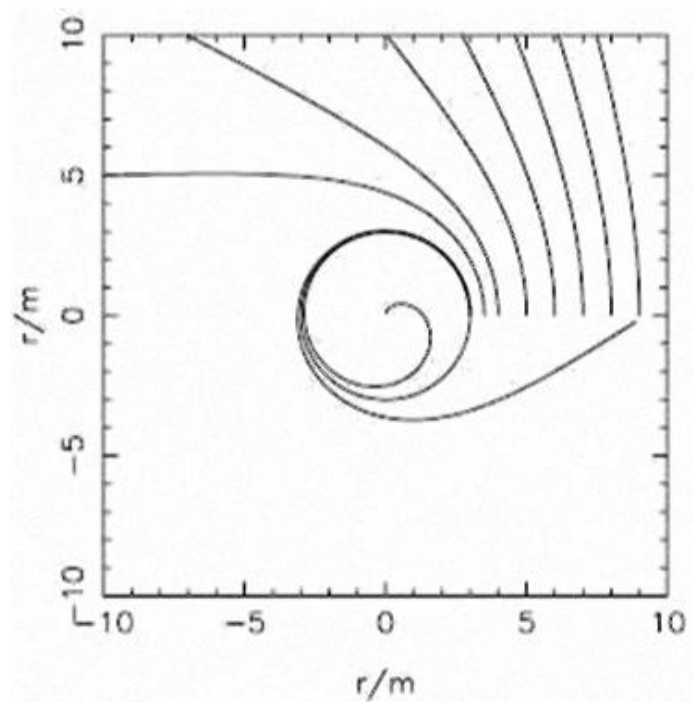
The distance between the various λ s varies in time by movements relative to each other. The directions of movements are being influenced by one another according to gravitational laws. The movement paths are being optically influenced for the outside observer by the curvatures of spacetime caused by the λ s themselves. This means that time slows down while relative space around a λ becomes smaller when the λ s are approaching each other. Time speeds up and relative space around a λ becomes larger when the λ s go from one another.

The λ is different than other particles in that respect that other particles consist out of multiple λ s while the λ itself is a singular particle. Each λ is a singularity (infinite curvature) on itself while other particles than the λ are a combination of multiple λ s and thus a system of multiple singularities.

The observed forces (strong, electromagnetic, weak and gravitation) have the same origin. The cause of these forces are because of the characteristics of a singular λ . The observed forces are in fact a sum of complex circular movements that come to exist when multiple λ s interact with each other.

Coding the dimensional basic

Figure 1: The tracks of two interacting λ s at different distances from each other.
(Original: Deflection of the tracks of a photon close to an object with a heavy mass.)



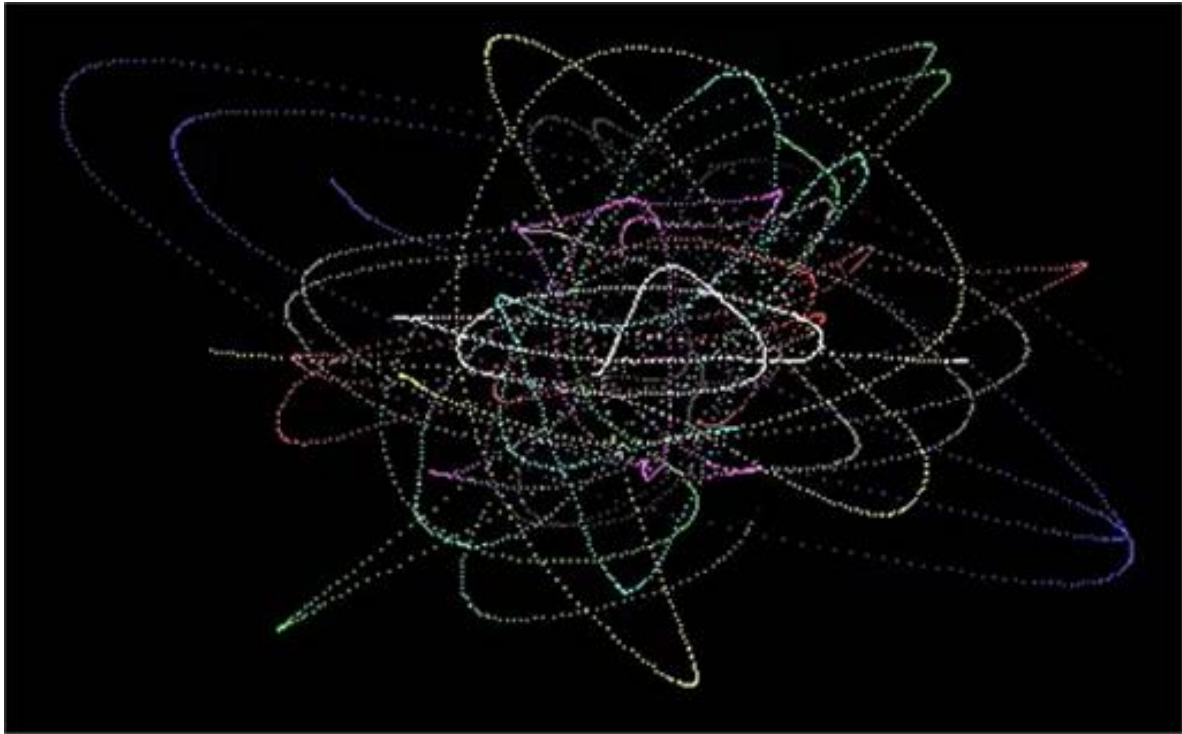
In figure 1 is shown how the movement tracks of photons react to the event horizon of a black hole. The same regularity applies to a binary black hole system. This is equal to the movement tracks of two λ s in respect to each other with the difference that the two λ s have no event horizon. These movement tracks are equal in behavior to Newton's laws of gravity. The Pauli principle is never violated because the λ s have no dimensions, they can approach each other, but can never touch each other.

On the basis of that information the Borland C computer model 'Newton' has been developed. This computer programmed model shows the movement tracks of λ s in three dimensional spacetime, in which the movement tracks of the λ s follow the gravitational laws. A three dimensional snapshot with nine interacting λ s is shown in figure 2.

The model 'Newton' is only accurate to a certain extent, it shows the movements of multiple interacting λ s in linear spacetime. In this model the by the λ s self deformed distances in

spacetime have not been implemented. The program 'Newton' gives the possibility to show an average time delay in video, thus making clear the principle of time delay.

Figure 2: Three dimensional view of calculated movement tracks of nine λ s during a random time.



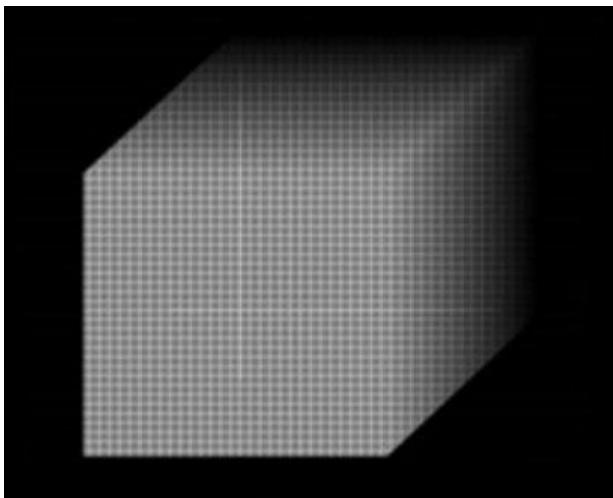
A second model that has been developed is the Borland C computer plot program 'Einstein'. This computer program has been developed to show how spacetime around a λ is being bent as seen by an outside observer, the extent of bending calculated according to formula (0).

Just like one λ as a singular singularity causes bending of spacetime because of an infinite curvature, a multitude of λ s will show a stronger bending of spacetime because of a sum of infinite curvatures. As Einstein made clear, we can speak of curved spacetime instead of linear spacetime. The more mass an object has, the more spacetime bends. One can say that invariant mass is the sum of the curvatures of a certain amount of λ s close to each other in respect to their surroundings. In case of for example three billion clustered λ s one can speak of three billion times infinite curvature. This makes it possible to isolate infinite numbers in comparison equations and thus clusters of λ s can be expressed as an absolute number. A cluster of a certain amount of λ s will have an absolute number of infinite curvatures. In this way one can speak of cluster A with X times infinite curvatures, while cluster B has Y times infinite curvatures. The infinities on both sides of the comparison can be done away with and only the absolute proportions of X and Y remain for the respective clusters. The curvature of a cluster of λ s with an absolute amount of λ s correlates with the invariant mass of an object and a certain extent of bending of spacetime.

The extent of bending of spacetime is calculated using formula (0), where the extent of curvature on a specific position of spacetime is being calculated. A bigger curvature means that spacetime is more bended, whereas a smaller curvature means that spacetime is less bended.

An example of this is shown in figure 3. In figure 3 the plot of a cube of spacetime is shown. The Einsteinian bending of a cube of spacetime is made visual. While figure 3a shows no bending of spacetime because of the absence of a λ , the bending in a cube of spacetime, and thus deformed distances for an outside observer, in figure 3b have been calculated according to formula (0) because of the position of a λ in the center of the cube of spacetime. At the center of the six surfaces of the cube of spacetime the distance to the λ is the smallest, for the outside observer it appears that that piece of spacetime is closer to the λ than it should be in linear (uncurved) spacetime, this because of the bending of spacetime, made visual by formula (0). Hence the pointy form of the corners of the cube of spacetime, there the distance to the λ is the biggest. Because of the bending of spacetime the distance is bigger for the outside observer than it should be according to a linear scale, this again made visual by calculating the extent of bending of spacetime according to formula (0). The closer spacetime is to a λ , the higher the curvature and the more spacetime will be bend.

Figure 3: Three dimensional calculated view of the bending of a cube of spacetime under the influence of a λ .



3a. Uncurved (linear) cube of spacetime.



3b. Cube of spacetime curved by the presence of a λ in the center.

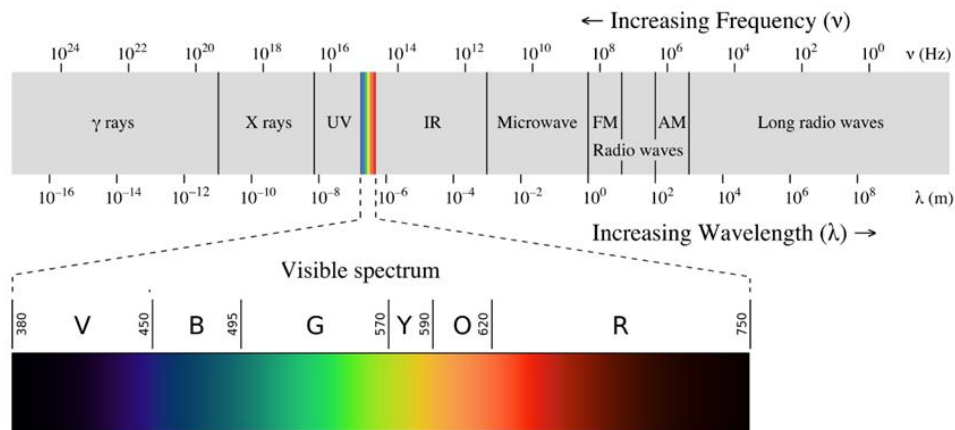
Concluding:

The Newtonian gravitational laws represent the straight movement paths as being caused by the bending of spacetime. Thus Newton's laws of gravity apply to the movement paths of the λ or a multitude of λ s. Both computer programs together represent the movement and character of the λ . The reality of the λ can be simulated by computer programs according to mathematical laws, taking into account the reality of formula (0) and the thereby caused

bending of spacetime. A third model, incorporating formula (0) in the cartesian coordinate system, splitting the calculation of the bending of space and the delay of time, should be able to simulate the universe as a whole, calculating and visualizing the λ movement tracks with Einsteinian bending of space and delay of time.

The photon

Figure 4: The electromagnetic spectrum.



When two λ -particles enter into the direct sphere of influence of each other's curvature, a strong interaction will be formed between the two. This is comparable to a star-planet combination such as the sun and the earth (illustration 1a). The difference is that the λ -particles are without a dimension and with an infinite curvature in the center (illustration 1b). This indicates that time, for the outside observer, infinitely slows down when the particles approach each other. So the combination of the two λ s has an enormous life span. The analogy of the curvatures around black holes is striking.

Illustration 1a: Earth in curvature field of sun.

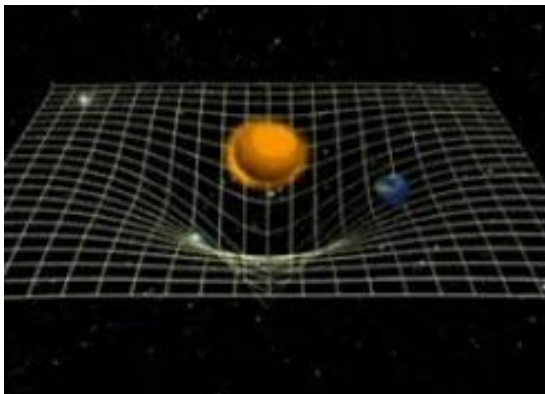


Illustration 1b: Depiction of curvatures 2- λ -particle.

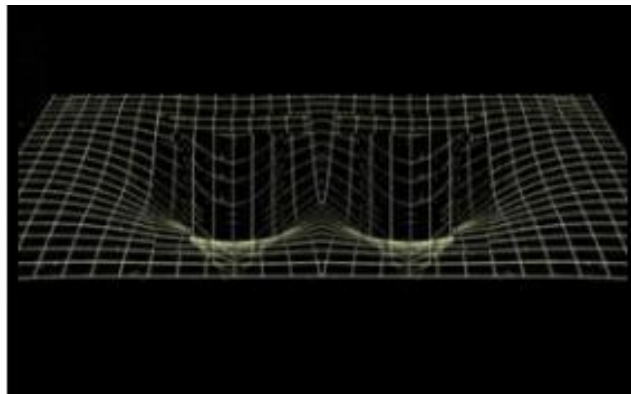


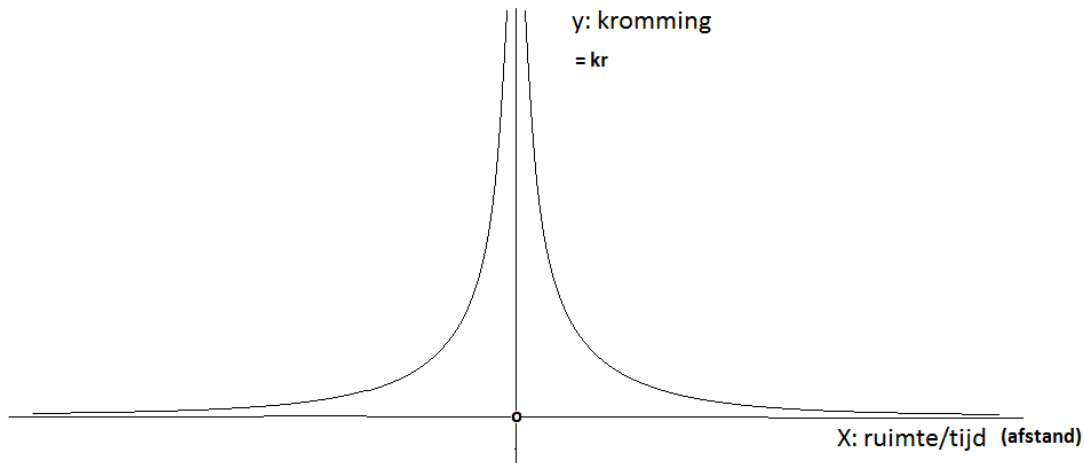
Illustration 1b is a representation of each particle from the electromagnetic spectrum. The hypothesis is that the 2- λ particle is a photon. Visible photons have a certain range for λ .

To calculate the curvatures around a single λ we can use a simplification of formula (0):

$$Kr = \text{abs} \frac{1}{x} \quad (1)$$

In the formula: Kr = curvature [m^{-1}], x = spacetime [m].

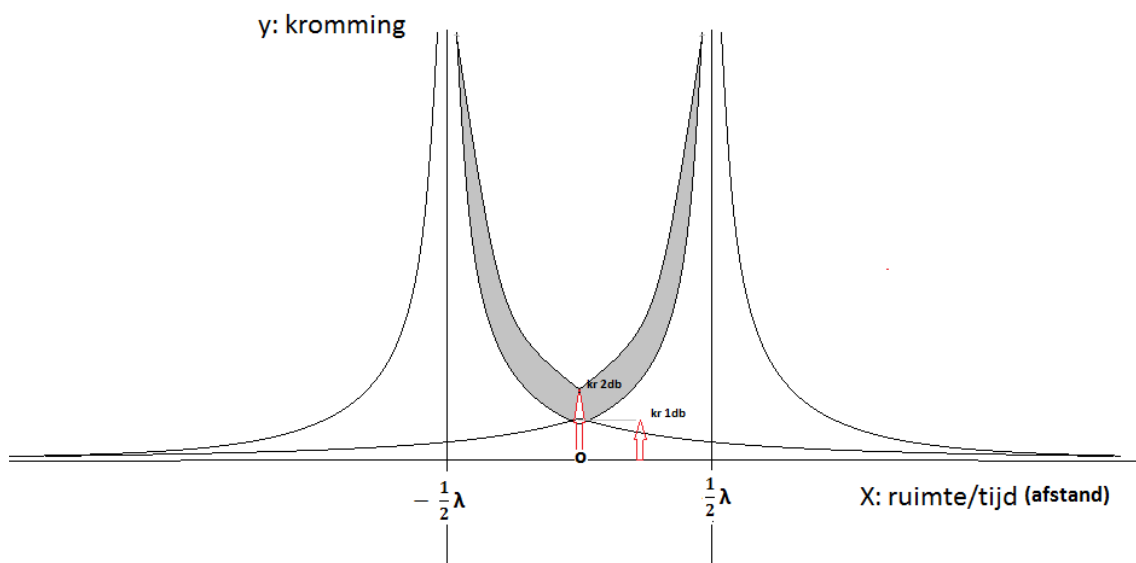
Figure 5: Absolute two dimensional schematic projection of the curvature strength around a λ where X is the distance in spacetime and Y is the amount of curvature (kr).



X : ruimte/tijd (afstand) = spacetime (distance), Y : kromming = curvature = kr

In figure 5 is shown that on the location of the λ (on $X = 0$) the curvature is infinite while the curvature becomes smaller when the distance to the λ enlarges. In figure 6 the graphic of a two- λ -system is shown. Two or more λ s result in a sum of curvatures on the spacetime surface between the λ s. In figure 6 this is made clear by marking resultant curvature in grey shade. One can say that the invariant mass of a two- λ -system is being caused by a stronger bending of spacetime between λ -particles.

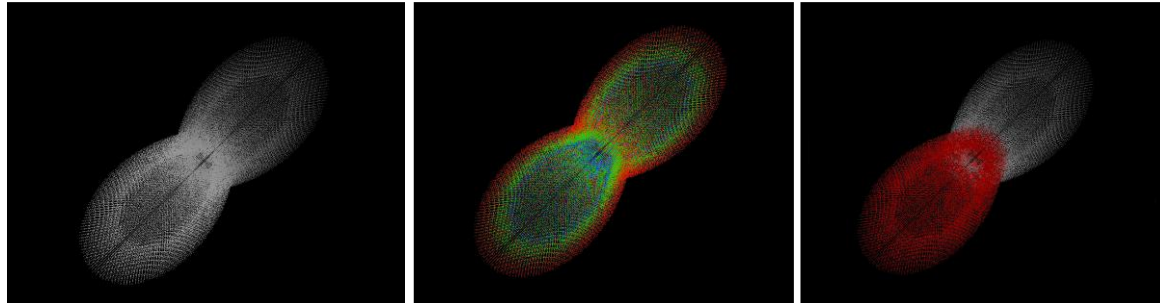
Figure 6: Absolute two dimensional schematic projection of the curvature strength around a 2- λ -system where X is the distance in spacetime and Y is the amount of curvature (kr).



$(X$: ruimte/tijd (afstand) = spacetime (distance), Y : kromming = curvature)

A calculation of curvatures that the observer can detect in a photon is shown in figure 7. The wavelength of the photon is equal to the distance λ between both particles. A schematic depiction of a photon and its movement in spacetime is shown in figure 8.

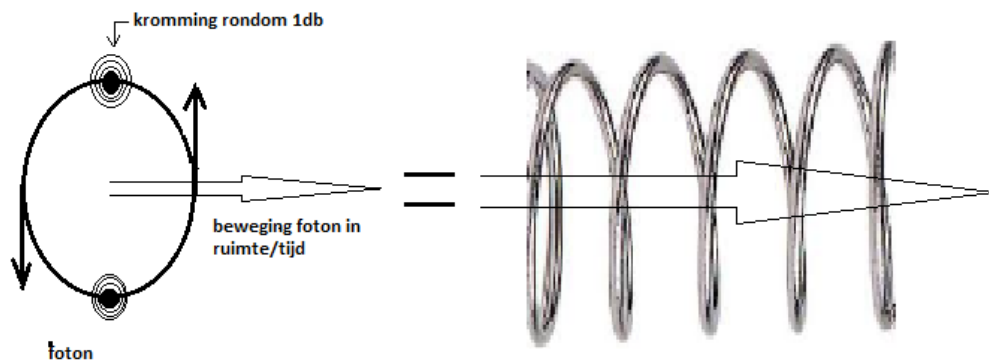
Figure 7: Three dimensional calculations of the curvatures of a 2- λ -particle (photon).



7a. Photon (greyscale).

7b. Photon (blue is high curvature, red is low curvature).
7c. Photon (each λ its own color).

Figure 8: Schematic projection of a photon.



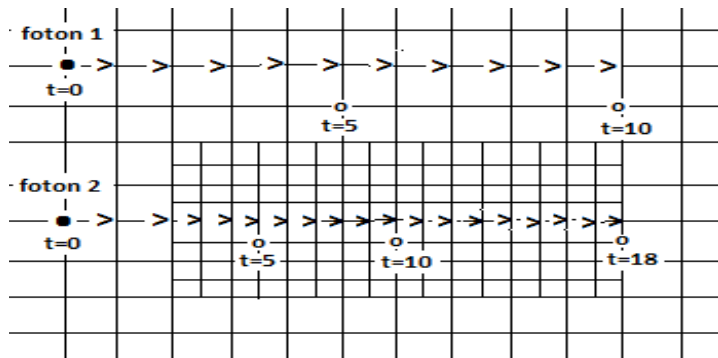
(kromming rondom 1db = curvature around 1db, beweging foton in ruimte/tijd = movement photon in spacetime)

A photon exists out of two λ -particles and therefore has an elongated form (three dimensional calculations; see figure 7). One can see this in polaroid glasses. The photons come through vertical (or horizontal) slits in the material of the glasses. The angle of the internal λ movement of the approaching photon (which can be 360 degrees, see figure 8) is right-angled to its forward movement in spacetime and makes that the photon comes through the slits of the glasses or not.

The internal rotation between the two λ s exhibits propeller-like dynamics - caused by the curvature field between them - the photon's propagation through spacetime can be considered a direct consequence of this internal geometric rotation. The photon then moves not because of an external field or from an inherent velocity, but because it rotates, and it rotates because of the local spacetime curvature. In this picture, the speed of light would not be a fundamental constant, but an emergent property.

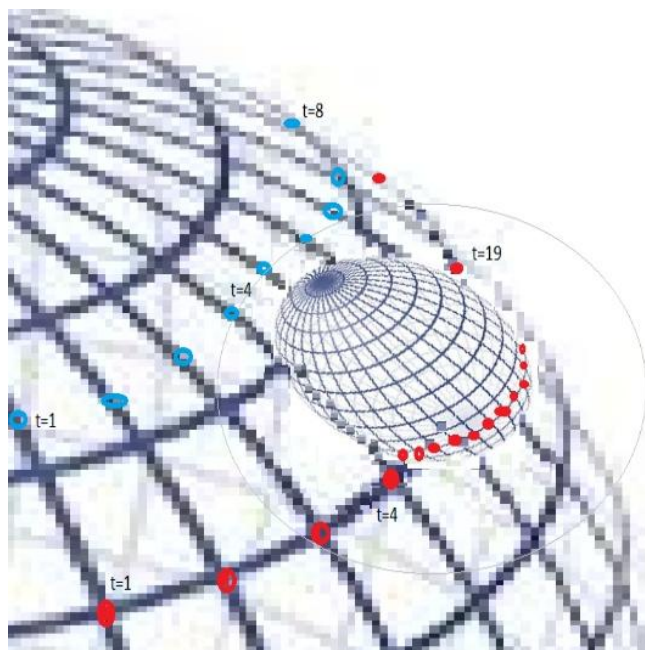
The speed of a photon in vacuum is $299\,792\,458\text{ ms}^{-1}$. In a medium like air, water or glass the speed will seem to be slower. This seems to be caused by higher curvatures close to particles the photon meets on its way through these materials. Figure 9 shows photons that have tracks through different curvature fields. Note that the photon 1 on t_{10} has a different position in spacetime than photon 2 on t_{10} . To the outside observer photon 1 seems to move faster. When you are traveling on the back of a photon you will not experience a delay, you will travel with constant speed.

Figure 9: Schematic view of two dimensional plane speed difference of two photons moving through different curvature fields as seen for an outside observer.



In a more realistic way the principle of apparent speed-delay is shown in figure 10. Here we see two photons traveling within the curvatures of a huge object. In this example the blue photon has no significant interaction with curvatures of the smaller object. We can state that the distance of the blue photon to the smaller object is relatively big. The red photon finds the smaller object in its track and is temporary caught by the curvatures of this object. The track of the red photon will give the outside observer the impression that the red photon is traveling slower than the blue photon but in fact it is traveling with constant speed.

Figure 10: Schematic view of three dimensional plane speed difference of two photons moving through different curvature fields as seen for an outside observer.



Cosmological consequences of the photon as a two- λ -system

It is clear that a moving 2- λ -particle – under the influence of a nearby object with an extreme curvature – will have a deflected track. This is in fact what is observed (see figure 1). If a photon on its track is influenced by curvatures caused by other particles, the photon will be brought out of balance. This means that the movement tracks in time of the internal two λ s will become centrifugal spiral shaped, i.e. the enlargement of the radius of its internal circular movement. Under the influence of extreme curvatures the photon will undergo a wavelength shift. We call this “the aging of the photon”. Because both λ -particles experience an enormous curvature via each other within the photon this is an extremely slow process for the observer. But during a trip through spacetime lasting many light-years (e.g. 10 billion light-years) the effect can be seen by the observer.

The cumulative effect of λ -curvatures across vast distances offers an explanation for large-scale cosmological observations. To date, the observed cosmic redshift in the universe has been explained mostly through the hypothetical expansion of the universe. The redshift is explained as a Doppler effect but it seems that the cosmic redshift is the result of the aging of the photon. This effect takes place when photons have traveled extreme distances (e.g. 10 billion light-years) in spacetime. As mentioned before, the aging of the photons is caused by the proximity of curvatures which the photon encounters in transit. As previously stated, these curvatures are present everywhere in the universe as λ s. The observed redshift is in fact a gravitational redshift. A direct conclusion could be that there is no such thing as an expansion of the universe. The observations of a seemingly accelerated expanding universe can be explained by the aging of the photon and thus there are doubts concerning the validity of the hypothesis of dark energy being responsible for the expanding of the universe at an accelerating rate.

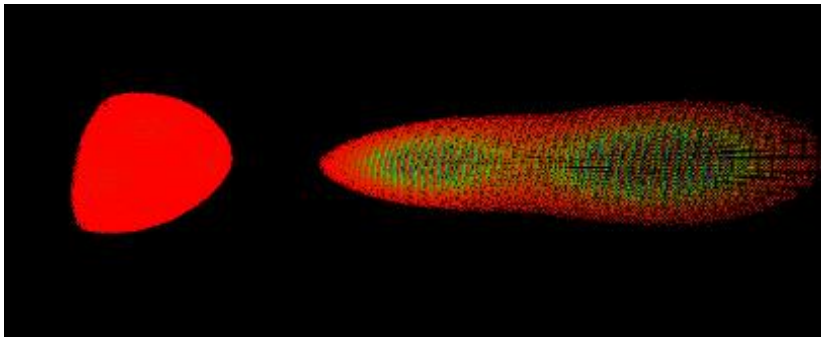
It is important to note that the large amounts of λ s are responsible for the observed presence of dark energy and dark matter. The λ s are in fact the sought after dark matter. This can explain the deviating speeds of galaxies. The movements in space can be explained in a Newtonian way. The by Einstein suggested cosmological constant in the theory of relativity is in fact a resumptive description of the presence of λ s. Einstein later on rejected his own suggestion on the basis of “Hubbles Law”. It seems that his suggestion indeed was right.

The λ plays a crucial role in the explanation of fluctuations in the spectrum of cosmic background radiation. The matter responsible has never before been observed. We believe that some types of the cosmic background are formed through the mutual interactions of the 1- λ -particles. This sometimes causes photons of completely different wavelengths to be formed, which together cause the pattern of cosmic background radiation.

Black holes

Under the influence of extreme curvatures in space the aging of a photon can accelerate greatly. This is observable near black holes (see figure 11). The closer the track of a photon to a black hole, the greater the aging. In fact, close to an event horizon (Schwartzschild scale) of a black hole the aging (gravitational redshift) is approaching infinity.

Figure 11: Three dimensional calculated view of the curvatures of a photon under the influence of an externally large curvature.

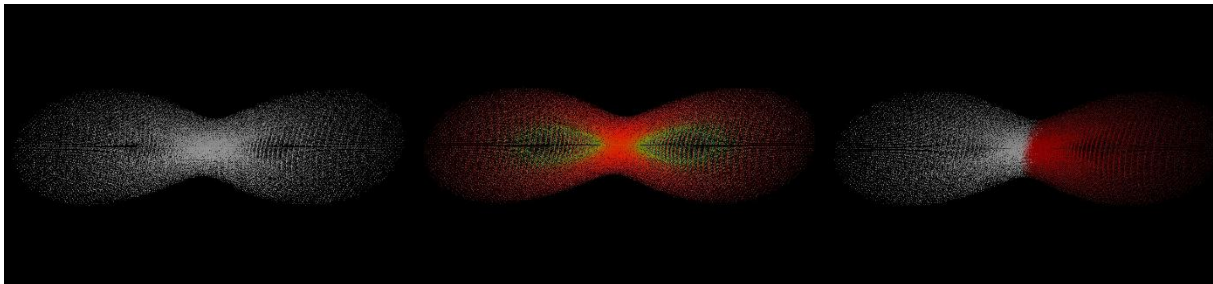


In our universe there is a limit to density of λ s, culminating in a black hole when the density gets beyond a boundary because of the relative high quantity of λ -particles. The curvature of the black hole depends on the size of the black hole and is a result of its internal λ quantity. Also here the Pauli principle will not be violated since the λ -particle will never get in the same position as another λ -particle. They can get close in a circling way while under the influence of each other's curvature. We can say that what appears to be instantaneous and linear in time and space for particles involved will appear to be a slow process for an outside observer. This means that within a black hole time will operate at a different level. The increasing curvature in the black hole system makes that time slows down for the outside observer.

In our universe photons have a certain range for λ (range approximately 1000 nm up to 1×10^{-3} nm). They are part of the electromagnetic spectrum where λ can have any value between zero and infinity. In our universe λ cannot be higher than the size of our universe. It might be that in a black hole (underneath the event horizon) the particles sequence of our universe repeats itself within a range of λ that is much smaller than the λ we can detect on earth. Theoretically λ in this black hole cannot be bigger than the size of this black hole. Imagine that if the λ range of the electromagnetic spectrum is not infinite in our universe, our universe is not infinite in spacetime and our universe may be a black hole for an observer above our universe. This observer is living in a universe where all the particles operate within a range that is much larger than ours. For this observer things on earth move rather slow. This observer could also tell if the black hole that our universe is would have a spin and the observer might observe that because of that spin there is a favorable direction in which λ s move, what might be the underlying cause for our universe to exist dominantly out of "right handed" particles (for example electrons instead of positrons).

Electrons

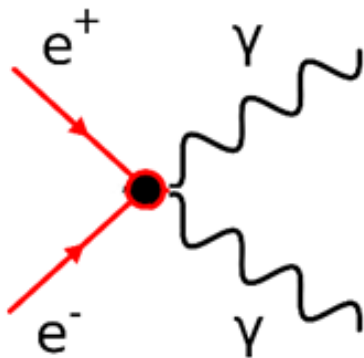
Figure 12: Three dimensional calculated views of the curvatures of an electron/positron. Respectively; greyscale is depth, curvature range where red represents relative low curvature while blue represents relative high curvature, individual curvature range for each λ .



When a photon with a speed of approximately $299\,792\,458\text{ ms}^{-1}$ moves through spacetime, the internal movement tracks of the $1-\lambda$ s are right-angled to this movement (figure 8). If a change of direction of this movement in respect to its internal λ movement takes place, this speed will be put in the spin of the photon. The hypothesis is that this photon with an extra internal spin is an electron.

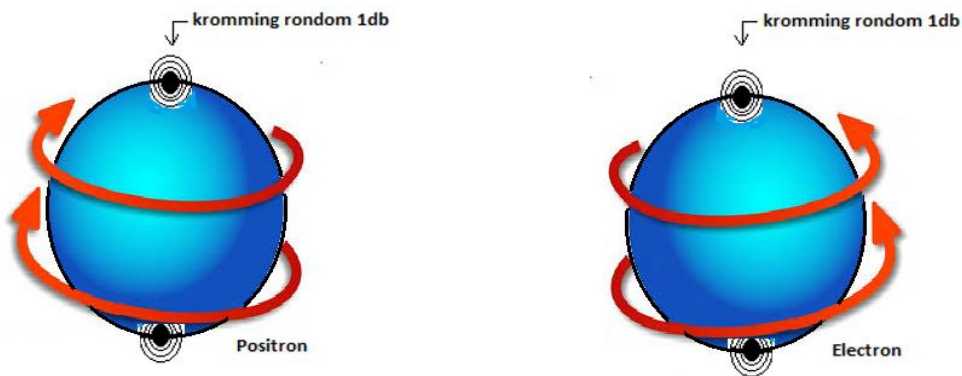
Observations have shown that a positron and an electron are annihilated which causes two gamma-photons to be released. This is depicted in the Feynman diagram below (figure 13).

Figure 13: Feynman diagram annihilation positron and electron.



At a confrontation between an electron and a positron a true annihilation does not take place. However, an “extinguishing” of both spins does take place in which the $2-\lambda$ -particles start to behave like gamma-photons. So this still refers to the same $2-\lambda$ -particles. The Feynman diagram can also be read in reverse. Two gamma-photons together form a positron and an electron. Each of the photons is made up of two λ -particles with only a rotation around the y-axis (see figure 8). The electron is a $2-\lambda$ -particle with an extra spin (towards the photon) around the x-axis (clockwise). The positron is also a $2-\lambda$ -particle with an extra spin around the x-axis, but counter-clockwise. This is depicted in figure 14. The photon is easy to imagine as a plate. The electron (or positron) can be imagined as a sphere.

Figure 14: Schematic depiction electron and positron.

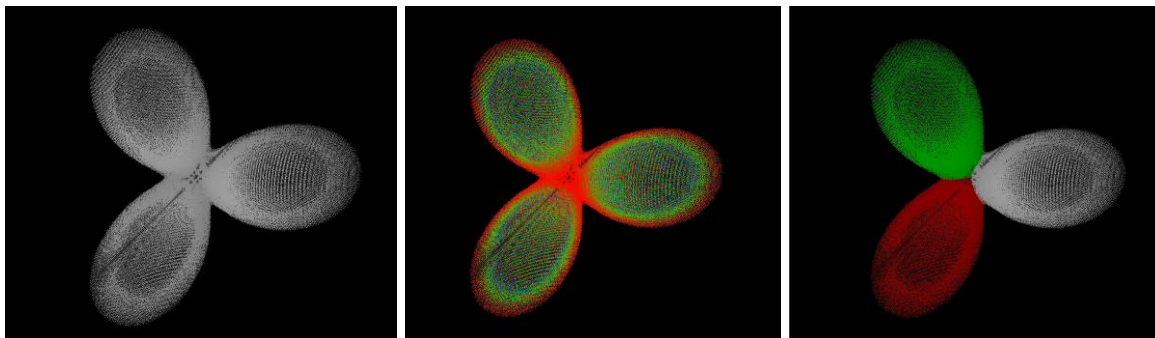


(kromming rondom 1db = curvature around 1db)

Quarks, protons and neutrons

Literature describes quarks as constituent particles. The quarks can occur in various ways. In a proton or a neutron one can see multiple quarks that are oriented up or down. A proton is known to consist of three quarks, two of which are up (2 Qu) and one down (1 Qd). Probably a quark is an interaction between three λ s. A calculation of curvatures as seen by the outside observer is shown in figure 15.

Figure 15: Three dimensional calculations of the curvatures of a quark.



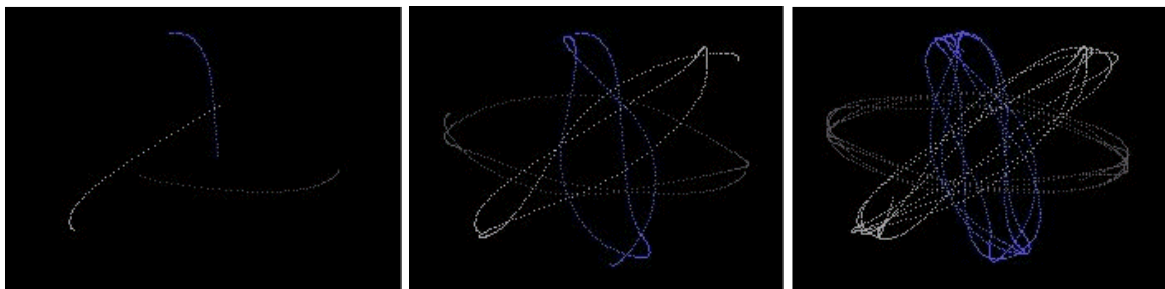
15a. Quark (greyscale).

15b. Quark (blue is high curvature, red is low curvature).

15c. Quark (each λ its own color).

Three dimensional calculated snapshots of the internal λ movement of a quark in spacetime can be seen in figure 16.

Figure 16: Three dimensional calculations of the internal movement tracks of λ s inside a quark.

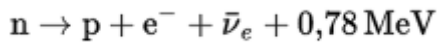


16a. Quark, time = 1 .

16b. Quark, time = 2.

16c. Quark, time = 3.

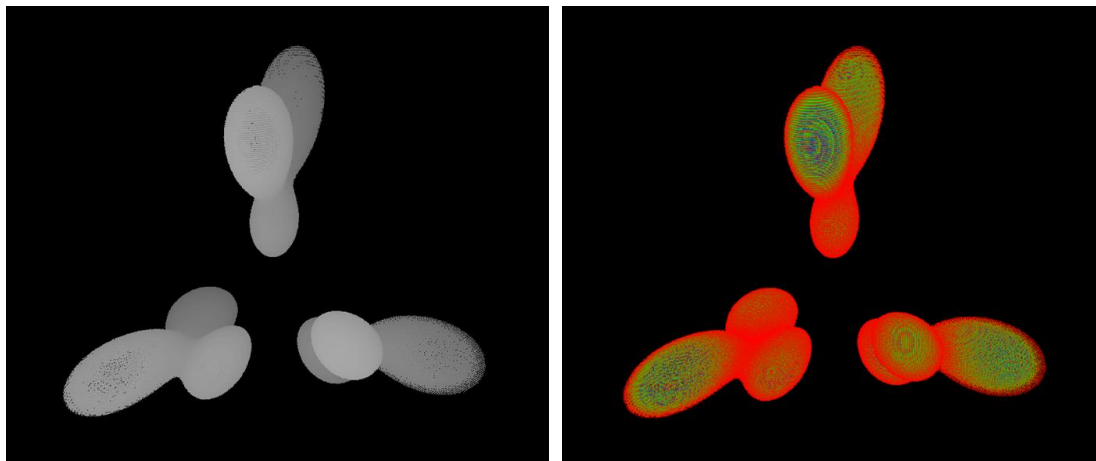
A neutron is unstable and rapidly dissociates into an electron, a proton and an electron-anti-neutrino.



From this comparison is inferred that a neutron loses a quark during its disassociation into a proton. The withdrawing quark (that consists of three λ s) is very unstable and will immediately disassociate into an electron (2- λ) and an anti-neutrino (1- λ). The anti-neutrino is in fact a 1- λ -particle that leaves the system of three (3- λ /quark) and in an ultra-short time displays an extra curvature in its immediate surroundings. This is observed as the anti-neutrino. The electron proves observable while the proton also forms.

One can conclude from this that a neutron consists of a foursome of quarks. Of these, two quarks are up and two quarks are down. This also explains the fact that, different from the proton, the neutron does not show a positively oriented field. The disassociation into a proton takes place during the expelling of a down quark. This will be further explained shortly.

Figure 17: Three dimensional calculated view of the curvatures of a proton as seen for an outside observer.

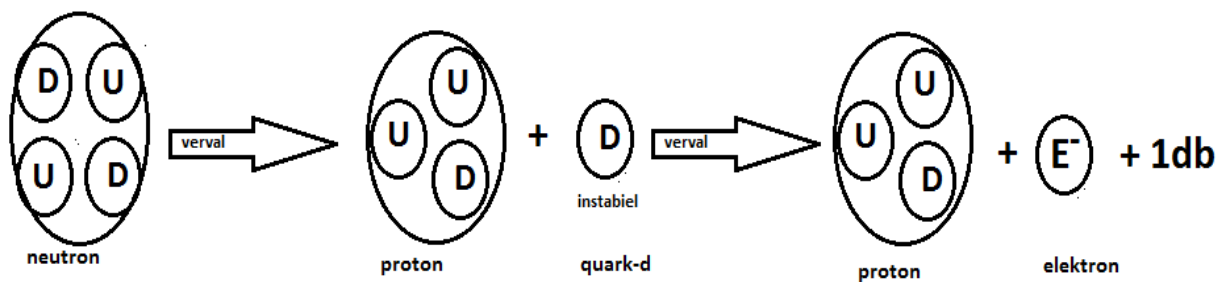


Experiments were performed that showed that neutron radiation consists out of uncharged particles with about the same mass as protons. This is an important reason that in the current insights the axiom is that both protons and neutrons consist of a threesome of quarks.

Thus a neutron consists of two up-quarks and two down-quarks (Qu, Qd, Qu, Qd). A calculation of the curvatures within a neutron is shown in figure 20. A proton consists of two up-quarks and one down-quark (Qu, Qu, Qd). A calculation of the curvatures within a proton is shown in figure 17.

Concluding: During the disassociation into a proton the following happens:

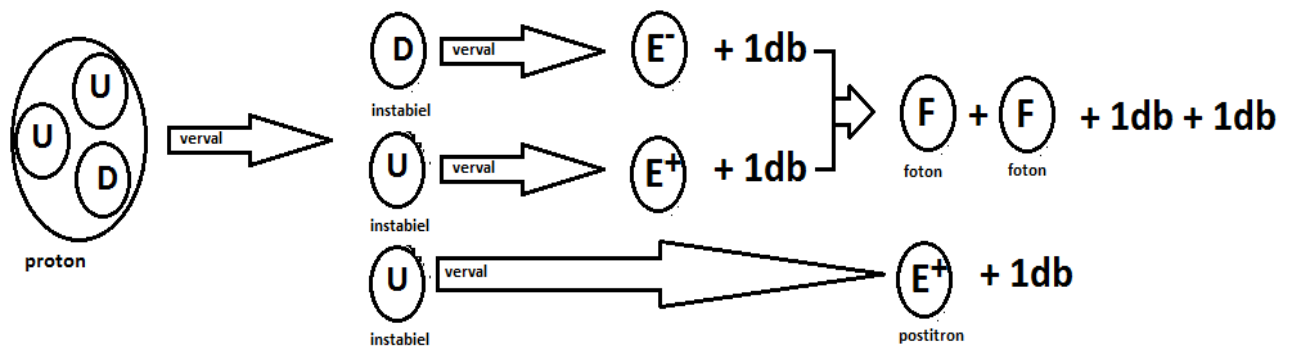
Figure 18: Disassociation neutron into a proton, electron and 1 λ .



(verval = decay, instabil = unstable)

In principle the proton is very stable. Yet it can be said that during the disassociation of a proton this will take place as follows:

Figure 19: Disassociation proton into a positron, 2 gamma-photons and 3x1 λ .

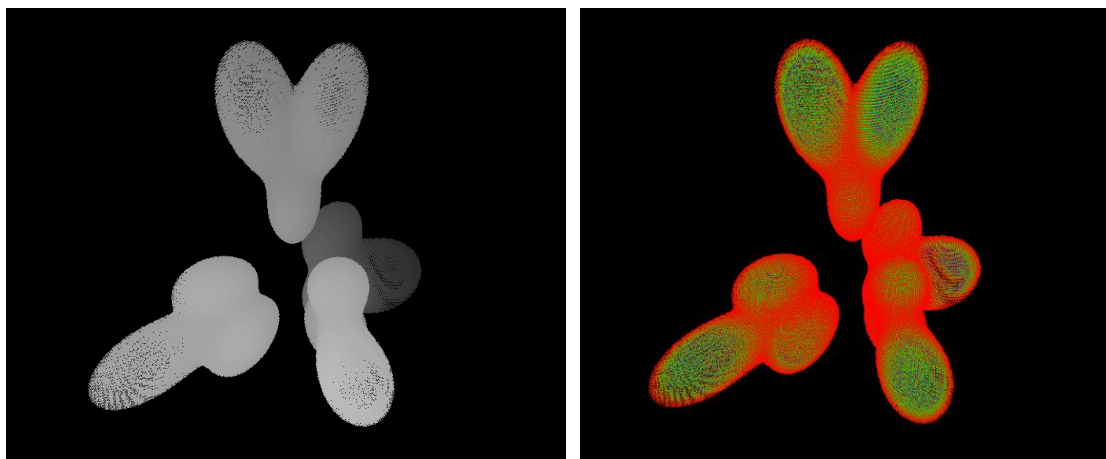


(verval = decay, instabil = unstable)

At a disassociation the proton will result in a positron, two gamma-photons and three 1- λ -particles. In an ultra-short time these 1- λ -particles will display an extra curvature in the immediate surroundings. These are observed as anti-neutrinos.

The described disassociation can in fact be observed by physicists. This provides the theory with evidence within the current observations.

Figure 20: Three dimensional calculated view of the curvatures of a neutron as seen for an outside observer.



Hawking radiation

Current insights on Hawking radiation

On Wikipedia the following can be found about Hawking radiation:

Hawking radiation is blackbody radiation that is predicted to be released by black holes, due to quantum effects near the event horizon. It is named after the physicist Stephen Hawking, who provided a theoretical argument for its existence in 1974, and sometimes also after Jacob Bekenstein, who predicted that black holes should have a finite, non-zero temperature and entropy.

Hawking's work followed his visit to Moscow in 1973 where the Soviet scientists Yakov Zeldovich and Alexei Starobinsky showed him that, according to the quantum mechanical uncertainty principle, rotating black holes should create and emit particles. Hawking radiation reduces the mass and energy of black holes and is therefore also known as black hole evaporation. Because of this, black holes that do not gain mass through other means are expected to shrink and ultimately vanish. Micro black holes are predicted to be larger net emitters of radiation than larger black holes and should shrink and dissipate faster.

In June 2008, NASA launched the Fermi space telescope, which is searching for the terminal gamma ray flashes expected from evaporating primordial black holes. NASA designed on 9 august 2007 the mission with a five-year lifetime, with a goal of ten years of operations. One of the key scientific objectives of the Fermi mission is: *"Search for evaporating primordial micro black holes (MBH) from their presumed gamma burst signatures [Hawking Radiation component]."*

Although Hawking radiation is not seriously questioned a steady evidence is not found by NASA or others.

Consideration 1

A photon consists of two λ -particles that are moving in each other's curvature (figure 8) and traveling through spacetime. A photon is influenced on its track by curvatures caused by other particles. Hereby the photon will be brought out of balance. Under the influence of extreme curvatures in space, the aging of a photon can accelerate greatly. This is observable near black holes. The closer the track of a photon to a black hole, the greater the aging. In fact, close to an event horizon of a black hole, the aging (gravitational redshift) is approaching infinity. In this process the photon will undergo spaghettification caused by extreme tidal forces. At a certain point the photon can no longer escape and is being captured by the black hole.

One can imagine that in the process of spaghettification there is a situation that the absorption is not yet an accomplished fact. In the "slow" process (for the outside observer) the 1- λ -particles within the photon are being separated as a wire. They are still entangled but the entanglement gets weaker. It is thinkable that at this moment one of the particles is

being captured and the other escapes from the event horizon. This will give a 1- λ -radiation which cannot be seen with the traditional methods. This radiation is not caused by vaporization of a black hole but is due to the devastating tidal forces of the black hole.

Sonic black hole in the laboratory

Jeff Steinhauer (Technion-Israel of Technology) claims to have created a sonic black hole to observe Hawking radiation and its quantum weirdness, all within the safe confines of his laboratory.

To recreate the effects of an event horizon Steinhauer used a laser to trap atoms in place on one side of the condensate and a second laser to create a step potential on the other side. The step potential acts like a waterfall, once atoms go over the limit they quickly speed up, travelling faster than the speed of sound and are considered as being inside the event horizon. Atoms that don't make it over the step potential move at subsonic speeds and are interpreted as being outside of the event horizon. The speed of flow in the superfluid mimics the gravitational pull of black holes.

Photons -particles of vibrational energy- escape (according to current insights) analogous to Hawking radiation.

Steinhauer's results are published in Nature Physics and show that (according to current insights) Hawking radiation is entangled. He measured the correlation function - a measure of how the properties of two particles in different positions are related - and found a high correlation for high-energy photons but low correlation for low-energy photons.

Photons equidistant from the event horizon were correlated and had equal and opposite energy, Steinhauer explained.

Particles are –according to Steinhauer- created out of nothing. According to Steinhauer it doesn't violate the principle of physics however, as the total energy balances out to zero.

“The entanglement verifies the quantum nature of the Hawking radiation,” the Steinhauer paper said. The next step, Steinhauer hopes, is to use artificial black holes to tackle larger problems like the information paradox or maybe even quantum gravity.

Consideration 2

In his experiment Steinhauer traps atoms and brings these atoms in an agitated state. Among these atoms there are neutrons. These neutrons can undergo the process as shown in figure 18 and figure 19. Seen from our perspective the complete decay of a neutron - through a proton- will end up in four photons [γ, γ]. There are two different pairs of photons. The pairs differ in characteristics. One pair of photons will be created instantly in the decay of a proton. This pair will be highly energized and highly entangled. The other pair is created when the electron that is created in the first step ($n \rightarrow e^- + 1\lambda + p$) finds the positron. The

positron is created in the second step ($p \rightarrow 2\gamma + 3\mu + e^+$). This photon pair will be less energized and less entangled.

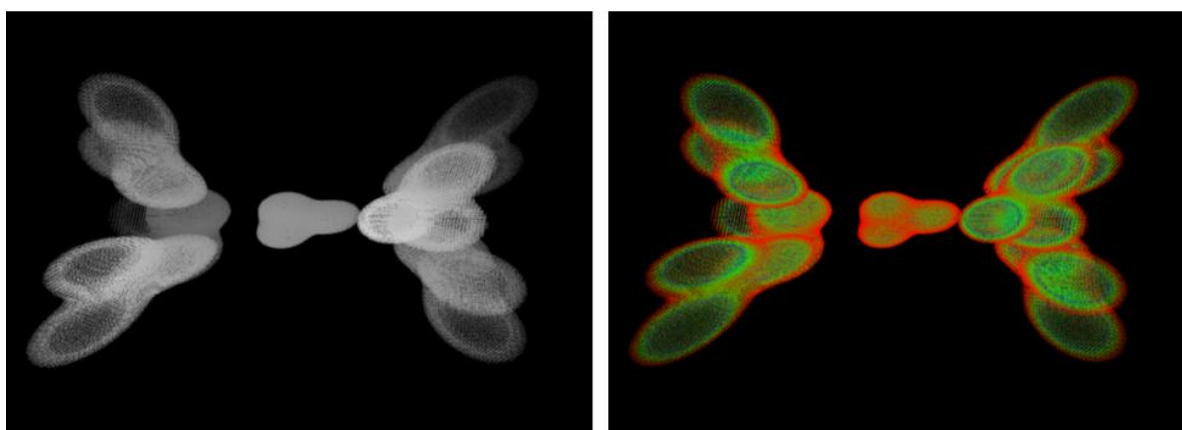
Our conclusion meets the observation of Steinhauer.

More complex particles

In more complex particles, the mutual interactions will become more and more complicated. These particles – rationalized from the basis – can be mathematically determined and simulated. Within these simulations we expect that the previously mentioned entanglements of particles can be explained. The entanglement is possible because particles (whether constituent or not) can be under the influence of each other's curvatures. This phenomenon can take place at very large distances. Such a situation will – caused by the relatively weak curvature – be unstable and experience a rapid disassociation. Because the entanglement is caused by curvatures, changes that one of the “partner-particles” experiences will instantaneously be experienced by the other “partner-particle”. Thus, there is an underlying, understandable reason for the observed transmission (no playing dice).

The principle of Einsteinian bending of spacetime is found within particles. When curvatures get extreme because of short distances within particles the outside observer will find that elements of the particle seem to come to a full stop. This seems the case for the outside observer but the inner particles (μ s, quarks, protons, neutrons) are still racing through spacetime with enormous speed.

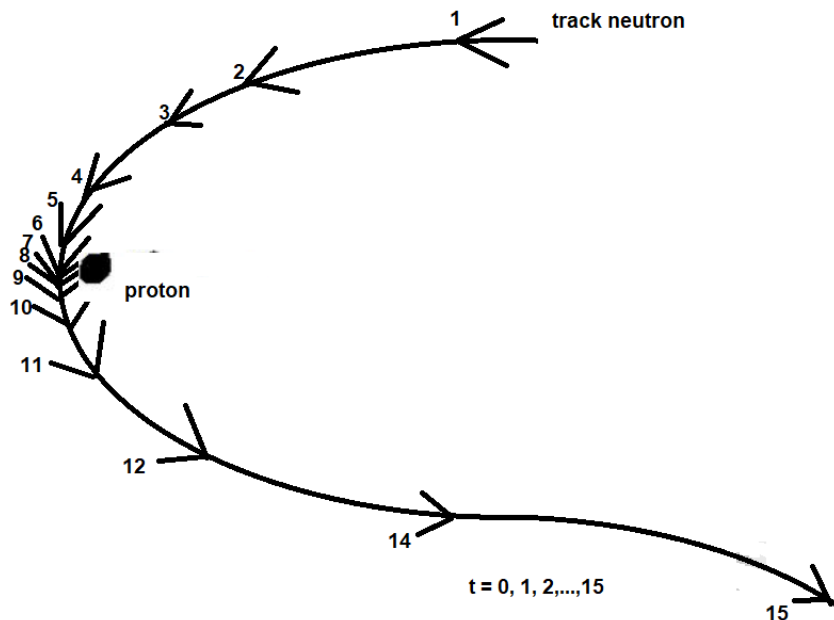
Figure 21: Three dimensional calculated view of the curvatures of a deuterium core as seen for an outside observer.



The core of a deuterium atom exists only out of one proton and one neutron. In figure 21 the curvatures of a deuterium core are shown. To the left the proton, to the middle/right the neutron. Remarkable is that the quark in the middle seems to be smaller than the surrounding quarks, this is the effect of a locally enlarged curvature of spacetime. The μ s,

the quarks, the proton and the neutron each have their own motion (rotation) in the deuterium nucleus. The proton and the neutron within their own complex movement tend to the configuration as shown in figure 21.

Figure 22: Schematic view of the trajectory of a neutron to a proton.



The timing within the described process is depicted in figure 22. In figure 22 the proton is held statically. The observer is theoretically situated on the proton. The proton and the neutron tend to circle in each other's curvature as shown in figure 22. In a Newtonian way they will approach each other as shown and then remove from one another. We can say that what appears to be instantaneous and linear in time and space for the proton and the neutron will appear to be a slow process for an outside observer. When the distance between the proton and the neutron becomes more narrow the movements seems to slow down for the outside observer. Movements seem to speed up again when the distance between the proton and the neutron gets bigger. The nearest point appears to be an "anchor" for the outside observer. Speed seems to come here to a full stop because of extreme curvatures. The half-life of the deuterium is unknown. The deuterium core appears to be stable but it is all a matter of perspective.

Positron Emission Tomography (PET) seen in a new light

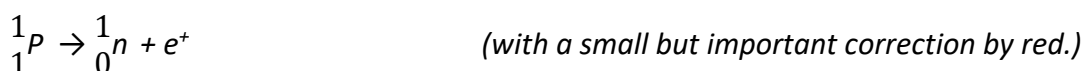
Positron Emission Tomography (PET) uses the principle of the so called annihilation of positrons with electrons while gamma-photons $[\gamma, \gamma]$ are released. These gamma-photons $[\gamma, \gamma]$ can be detected with a scanner giving information about location and appearance of specific organs.

The necessary positrons can be given by radioactive decay of certain atoms like Carbon-11 and Oxygen-15.

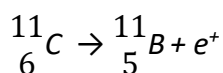
Current insights

The case Study “Positron Emission Tomography (Last updated 11:48, 12th June 2016)” gives the following description:

The emission of a positron is represented by:



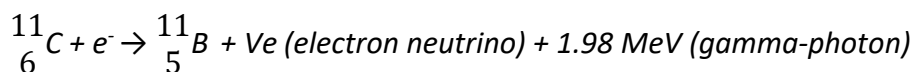
This shows that the positron (represented here by e^+) speeds out of the nucleus while the neutron stays inside the nucleus. Consider the following nuclear reaction that is common in PET scans of the brain where carbon-11 is used as the tracer molecule.



Notice that in this example of positron emission, the nuclide changes into a different element and as it gives off a positron particle, the atomic number is lowered by one, but the mass of the new element stays the same as the carbon that has decayed.

Wikipedia on this subject:

Carbon-11 or C^{11} is a radioactive isotope of carbon that decays to boron-11. This decay mainly occurs due to positron emission; however, around 0.19–0.23% of the time, it is a result of electron capture. It has a half-life of 20 minutes.



Carbon-11 is commonly used as a radioisotope for the radioactive labeling of molecules in positron emission tomography. Among the many molecules used in this context is the radioligand [^{11}C]DASB (labeled with carbon-11).

Decay of protons

Positron Emission Tomography (PET) uses –as assumed in the published articles (Case Study: Positron Emission Tomography (Last updated 11:48, 12th June 2016))– the transition of protons to neutrons.

On Wikipedia one can find that there is currently no experimental evidence that proton decay occurs when a proton is on its own, while for a neutron one can find that the decay of the free neutron is possible.

Wikipedia: $n^0 \rightarrow p^+ + e^- + \nu_e$

ν_e in the formula is a little more or less 0.78 MeV ($1,25 \cdot 10^{-13}$ Joule), $1 \text{ eV} \approx 1,60 \cdot 10^{-19}$ Joule.

There are claims of the transition of protons to neutrons within the decay of more complex particles. In many cases there is a relation with Positron Emission Tomography. Relevant in this case is the decay of C^{11} and O^{15} .

Consideration

According to our theory the transition of a solitary proton to a neutron is complex and not to be expected. Following our equation the organization of matter needed in a reverse reaction by which a proton changes into a neutron does not lead to the forming of gamma-photons $[\gamma]$.

The equations are given in figure 18 and figure 19 in the chapter 'Quarks, protons and neutrons'. According to the theory only the decay of a proton can lead instantly to the appearance of gamma-photons $[\gamma]$.

The theory suggests different mechanisms for Positron Emission Tomography (PET). First an explanation for the decay of Carbon-11 is given, then an explanation for the decay of Oxygen-15 is given.

Carbon-11

Carbon-11 is an isotope of carbon, frequently used in positron emission tomography, or PET imaging. It has 6 protons, 5 neutrons, and 6 electrons. It has a half-life of 20 minutes.

Decay C^{11} (2 atoms)

The first C^{11} gives:
$${}_{6}^{11}\text{C} \rightarrow {}_{6}^{10}\text{C} + {}_0^1\text{n} \quad (\text{a})$$

C^{10} is not stable/half-life 20 sec*, ${}_0^1\text{n}$ is used in (c).

C^{10} out of (a) decays further as:
$${}_{6}^{10}\text{C} \rightarrow {}_{5}^{10}\text{B} + {}_1^1\text{p}^{\nearrow} \quad (\text{b})$$

B^{10} is stable*, possibility: $[p \rightarrow 2 \gamma + 3 \mu + e^+]$, in that case the proton will not be seen, the e^+ will follow (d) .

A second C^{11} gives:
$${}_{6}^{11}\text{C} + {}_0^1\text{n}^{\leftarrow} \rightarrow {}_{5}^{11}\text{B} + 2 \gamma^{\nearrow} + 3 \mu^{\nearrow} + e^+ \quad (\text{c})$$

In (c): $[p \rightarrow 2 \gamma + 3 \mu + e^+]$, ${}_0^1\text{n}$ is delivered by (a), B^{11} is stable*.

The e⁺ out of (c) finds an e⁻: $e^+ + e^- \rightarrow 2 \gamma$ (d)

Total equation: $2x \frac{11}{6}\text{C} + e^- \rightarrow \frac{10}{5}\text{B} + \frac{11}{5}\text{B} + 4 \gamma + 3 \mu^+ + \frac{1}{1}\text{p}^+$ (e)

Possibility: [$p \rightarrow 2 \gamma + 3 \mu^+ + e^+$], in that case the proton will not be seen, the e⁺ will follow (d).

Oxygen-15

Oxygen-15 is an isotope of oxygen and is also frequently used in positron emission tomography, or PET imaging. It has 8 protons, 7 neutrons, and 8 electrons. It has a half-life of 122 seconds.

Decay O¹⁵ (2 atoms)

The first O¹⁵ gives: $\frac{15}{8}\text{O} \rightarrow \frac{14}{8}\text{O} + \frac{1}{0}\text{n}$ (f)

O¹⁴ is not stable/half-life 70 sec*, $\frac{1}{0}\text{n}$ is used in (h).

O¹⁴ decays further as: $e^- + 1 \mu^- + \frac{14}{8}\text{O} \rightarrow \frac{14}{7}\text{N}$ (g)

In (g) [$e^- + 1 \mu^- + p \rightarrow n$], N¹⁴ is stable*.

A second O¹⁵ gives: $\frac{15}{8}\text{O} + \frac{1}{0}\text{n} \rightarrow \frac{15}{7}\text{N} + 2 \gamma + 3 \mu^+ + e^+$ (h)

In (h): [$p \rightarrow 2 \gamma + 3 \mu^+ + e^+$], $\frac{1}{0}\text{n}$ is delivered by (f), N¹⁵ is stable*.

The e⁺ out of (h) finds an e⁻: $e^+ + e^- \rightarrow 2 \gamma$ (i)

Total equation: $2x \frac{15}{8}\text{O} + 2e^- \rightarrow \frac{14}{7}\text{N} + \frac{15}{7}\text{N} + 4 \gamma + 2 \mu^+$ (j)

The given half-lives* are taken from: <http://periodictable.com/Isotopes/007.15/index2.p.full.prod.html>.

Electromagnetic fields

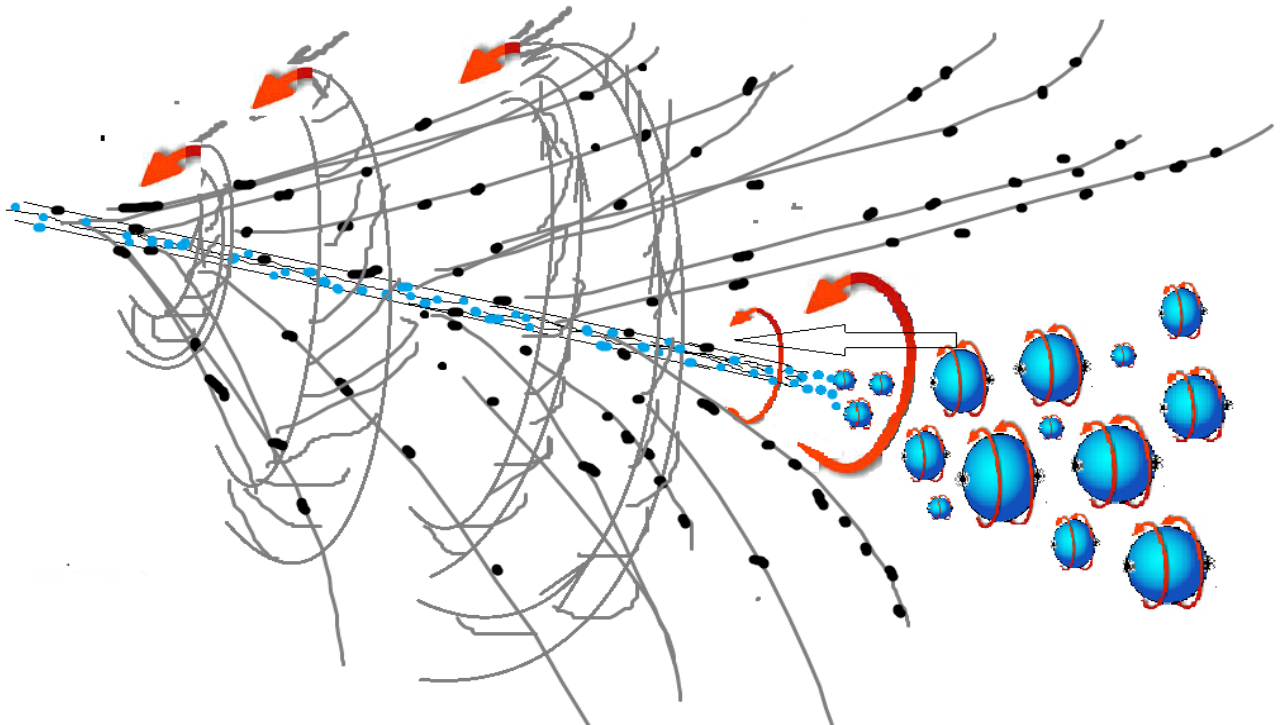
Magnets and magnetism are known for centuries. The study of magnetic fields seems to have begun in 1269 when French scholar Petrus Peregrinus de Maricourt mapped out the magnetic field on the surface of a spherical magnet using iron needles. He also clearly articulated the principle that magnets always have both a north and a south pole.

Although magnetic fields are well known and well-studied there is no explanation for the existence of field lines. Field lines can be made clear when using iron fillings randomly dropped on paper that is covering a magnet. When one shakes the paper in a gentle way the

field pattern will occur. The pattern is caused by the bending of the movement tracks of $1-\lambda$ -particles in a discrete pattern around the magnet. This is caused by the curvature of electrons that circle in discrete orbitals around the nuclei of the used (electro)magnet. Due to the influence of curvatures of specific atoms that are present in the material needed to create the magnet the field lines are created.

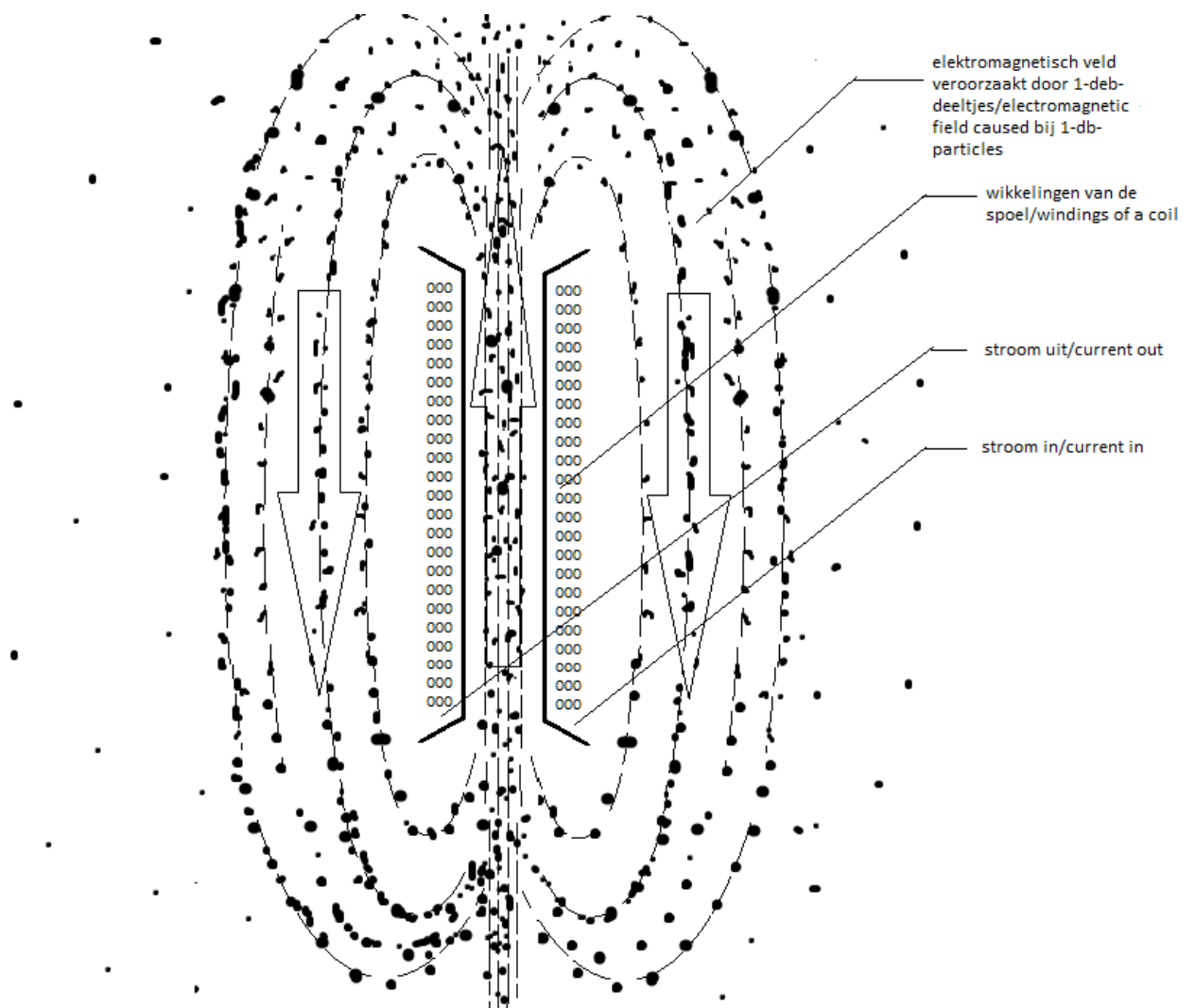
Electromagnetic fields around an energized wire behave like fluids within a centrifugal pump. If the fan of a centrifugal pump begins to rotate the fluid within the fan will get a tangential speed (= speed in the direction of the periphery). The centrifugal force that hereby arises makes the fluid being pushed to the outer periphery of the fan. In this the mechanical energy (the rotation of the fan) is being converted into potential and kinetic energy. In analogy to, the electrons (who all have a like-minded spin) will be hurled to the outer periphery of the wire. On the outside of the wire the curvatures caused by the electrons will be large. Through these curvatures the $1-\lambda$ -particles will be sucked in. This causes a whirlwind of $1-\lambda$ -particles which will rotate around the energized wire. This causes the electromagnetic fields with their attractive force. This process is depicted in figure 23.

Figure 23: Schematic view of electromagnetic fields around an energized wire.



By winding an energized wire in a coil the electromagnetic forces are being cumulated, this resulting in the fields as observed around an energized coil. This process is depicted in figure 24. When positrons are sent through a wire the fields will show an opposite direction with respect to the fields caused by electrons.

Figure 24: Schematic view of electromagnetic fields in and around an energized coil.



Memorandum about quantum field theory (QFT)

Most physicists approach questions around the electromagnetic according to the quantum field theory. The quantum field theory (QFT) is meant to setup a quantum mechanical theory of the electromagnetic fields. With the QFT quantum degrees of freedom in space (particle states) are being indexed. Then symmetric quantum states together form a quantum field. The particles behave identical and together are the cause of a resulting quantum field (electromagnetic field). The current quantum field theory is mathematically not rigorous. The last decades several attempts have been made to place quantum field theory on a solid mathematical base by formulating a set of axioms for it. Finding the right axioms still is an open and difficult mathematical problem. This is one of the Millennium prize problems.

The curvatures around a particle are fundamental for arousing an electromagnetic quantum field. There is a symmetry that creates a symmetric field as described in the QFT. There is

however little reason for the description and the use of symmetric (bosonic) or anti-symmetric (fermionic) states. The arousing of an electromagnetic field can completely and fully be understood from the perspective of Einstein's teachings of curvatures. The field will occur when particles like electrons accumulate at a specific position or in a specific movement. This will instantly lead to the attraction of λ -particles by which a quantum field can be observed.

Acknowledgement

The λ was devised by Gerhard Jan Smit during the years 1986 to 1993. He shared the theory of the λ , the character of dark matter, electromagnetic radiation, electrons, quarks, curvature phenomena of complex particles, the relative variable speed of light through various curvature fields, the aging of a photon, the improbability of the hypothetical expansion of the universe, the λ 's responsibility for the movement of galaxies, its responsibility for the cosmic background and the properties of black holes on the 7th October 2016 with Jelle Ebel van der Schoot.

Further deductions of the theory applying to electrons, positrons, the cosmological constant and the deuterium core were developed jointly. Jelle Ebel van der Schoot has posited the theory of the proton and the neutron and their decay. In December 2016 Gerhard Jan Smit has calculated and described the properties of a deuterium core while on the 7th of January 2017 Jelle Ebel van der Schoot has found and described an explanation for electromagnetic fields, both starting from the then present theory. All this has resulted in the present article.

Figure 1 is from: "Presentation black holes", John Heise, Utrecht University. Illustration 1a is from Building Blocks of the Universe, Len Zoetemeijer. Illustration 1b is a derivative of illustration 1a. Figure 4 and figure 13 come from the internet, the other figures are own production. Figure 2 and figure 16 were created using the program 'dbmove.bas'. The representations of the curvatures of a cube of space, photons, electrons, quarks, protons, neutrons and the deuterium nucleus were made using the plot program 'einstein.cpp'. Gerhard Jan Smit developed both programs in the computer languages Microsoft Quick Basic and Borland C++ under the Microsoft Disk Operating System, respectively in 1995 and 1996.

Some of the content of the chapter "Observed conflicts within quantum mechanics" is based on "Review of Roland Omnès, The Interpretation of Quantum Mechanics", William Faris, November 1996. Insights about the universe are taken from the books "The Point Omega", John Gribbin, 1988 and "Galaxies in the Universe", L.S. Sparke and J.S. Gallagher III, 2007.

Knowledge is easily found using external sources on the Internet.

The information on protons, neutrons, quarks and particle decomposition is general information that can be found on Wikipedia. References regarding the relevant topics can be followed through the Internet page [DB Physics External Resources](#).

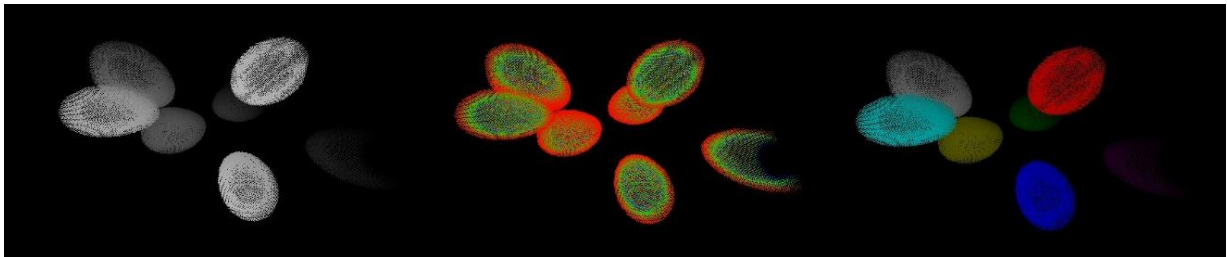
We express special gratitude to Democritus, Newton, Einstein, and for the remainder to God, who does not play dice.

Authors: Gerhard Jan Smit, Jelle Ebel van der Schoot, Nijmegen, The Netherlands.

© 2016, Version 1.0 legally registered 21st November 2016,

Version 2.0: 21st Januari 2018, Revision 1st April 2021.

Figure 25: Curvatures dark matter: Seven λ s. Respectively; greyscale is depth, curvature range where red represents relative low curvature while blue represents relative high curvature, individual curvature range for each λ .



Website: www.dbphysics.com Contact: info@dbphysics.com

Forum

Question: There can be circumstances that particles like electrons accumulate on a specific position. How do you make sure that the Pauli principle is not violated in this case?

Answer: It is an empiric fact that electrons can accumulate. The Pauli principle will –in our opinion- not be violated since the db-particle will never get in the same position as another db-particle. They can get close in a circling way while under the influence of each other's curvature. What appears to be instantaneous and linear in time and space for two particles will appear to be a slow process for an outside observer. This is the case when particles get really close like quarks within an atom. The increasing curvature in the system makes that time seems to slow down for the (human) observer. The Pauli principle is never violated.

Question: In the article you state: “this model constitutes a good candidate for a new foundation to represent the observed particles and forces. The short distance force (weak) and the long distance forces (strong, electric and gravitational) can be explained from the described curvatures”. Gluons are massless. The Higgs mechanism only adds mass to weak interaction particles (Z/W-bosons). How do you take that into account?

Answer: We are of the opinion that the strong and the weak forces have exact the same origin. The strong forces within an atom (for instance within a proton) where the quarks have found an anchor-point is stable due to the short distances. Within the atom for the outside observer time has slowed down. That is why the position of the quarks in the atom

seems stable. It is only a matter of perspective. In a set of molecules (for instance water) where the distances between the different molecules is such that the molecules are within a reasonable influence of each other's curvature also a stability will be achieved. The molecules will stay together in a structure but obviously the situation is not stable as the (human) observer can observe.

About "the massless gluon". We are not convinced of the existence of gluons. But we always argue within our theory. We admit that we lack knowledge but in some way that can be an advantage. Maybe you will disagree with that. I am sure you will.

Question: If particle and wave is the same thing u should also write zero point wave.

Answer: A particle consisting of multiple db's will imprint an extra curvature on the spacetime surface between the db's. In this a particle is not a wave, it is a cluster of more than one interacting db's. The wave property it possesses are its internal db-movement tracks in time. These movement tracks in time can be described as a wave function. A singular db does not have a wave property.

So particle and wave is not the same thing.

Although it can be said that in case of a multiple db particle the extra curvature imprint on spacetime is a wave function in itself, so then the particle equals that wave function it exhibits in time. One can say that the multiple db particle is in a sense the fluctuating spacetime surface and is in this case the wave.

Question: How will this particle be affected by time?

Answer: The higher the curvature of the multiple db particle, the slower its internal movements will seem for the outside observer. Internal time dilation because of the relative strong bending of spacetime.

Question: Is it similar to a black hole?

Answer: The singular db has a property that is almost similar to a black hole. The db has a black hole like curvature imprint on its surrounding spacetime. The difference is that the db particle has no spatial dimensions (length, width, height) and a black hole does.

The db is a singularity, the curvature of the particle is infinite (or so to say, spacetime is infinitely bended) on the location of the db.

Question: It is hard to imagine particles without spatial dimensions. It is almost similar to energy. For my own knowledge, if a black hole reaches singularity why does it emit radiation. Could this mean that when something reaches singularity it changes its dimension to energy?

Answer: In the db model only the db itself has an infinite curvature and is the only singularity that exists. All macro structures, from elementary particles to black holes, exist out of those

singularities but are never a singularity on its own. They can get very high curvatures on the spacetime surface between the db particles but always a fraction of infinity.

So in our universe nothing but the db ever is a true singularity.

Energy is always a resultant power of miscellaneous variables and is a property of spacetime. The more spacetime bends within the multiple db particle, the more energy the particle contains. But it will never contain an infinite curvature on the spacetime surface between the db's so even if singularity changes its dimension to energy, it will not happen since none of the multiple db particles will ever reach singularity, not even a black hole.

A black hole has an enormous curvature on its event horizon, the more mass, the more spacetime bends, but it is limited in its amount of bending of spacetime. The bending of spacetime on the event horizon is such that it destructs all traditional known particles, including photons, but the curvature experienced by the particles will not be infinite, but is dependent on the internal db quantity of the black hole which lead to a specific curvature strength at the event horizon of the black hole.

A black hole can emit various types of radiation. Within the theory of the db there is of course db-radiation. This can occur in various ways. Db's, if under the right angle and right speed, can leave the black hole system and one could say that it is db radiation. Furthermore all types of radiation will be emitted in the process of decomposing particles that get to near to the event horizon of a black hole. Those particles are ripped apart due to the tidal forces of the black hole. The elements of the decomposed particle that can escape the event horizon will be the observed radiation.

Question: So which dimension does db particles exist in??

Answer: The db exists in 3 dimensional spacetime where it has a location and on that location the curvature and thus the bending of spacetime are infinite.

Question: The equation written about db is it correct? And did u discuss on physics forums and what do they think? I am not good in math.

Answer: The equation is correct, but also an assumption. It has been derived step by step through deducing and writing computer algebra to support the theory. In the end the only logical conclusion for the curvature around a db is according to formula (0).

The theory has not yet been discussed on many fora because it isn't taken all to serious. It seems to defy the theories of general relativity and quantum mechanics, which is not the case. It just lays down a deeper, more fundamental explanation for the observed forces and particles, in sync with observations in various fields of physics as described in the article.

The theory is like putting on glasses to see even sharper into the micro world than before.

Addendum - Computer program sources

The addendum shows the source codes of the Borland C computer programs 'Newton' and 'Einstein', preceded by a MS Quick Basic example of μ movement analysis. The addendum finishes with a patent for an analogue computer chip.

The MS Quick Basic model and the first Borland C model are an interpretation of the Newtonian gravitational laws. In this first Borland C model the paths of μ s are calculated using the traditional Newtonian laws.

The second Borland C model is a three dimensional snapshot of curvatures of spacetime caused by the most elementary possible particles (μ s) as derived from Einstein's ideas.

1) db movement analysis (MS Quick Basic)

```
REM (C) G.J. Smit, Nijmegen, Nederland
REM This software is published under the GNU General Public License v3.0
REM www.dbphysics.com
REM Program purpose: db movement analysis
KEY(1) ON: ON KEY(1) GOSUB afrondschoonscherm
KEY(2) ON: ON KEY(2) GOSUB andermode
KEY(3) ON: ON KEY(3) GOSUB nieuwecoordinaten
KEY(4) ON: ON KEY(4) GOSUB windowgrootte
KEY(5) ON: ON KEY(5) GOSUB sterktezwaartekracht
KEY(6) ON: ON KEY(6) GOSUB nieuwaantaldeeltjes
KEY(7) ON: ON KEY(7) GOSUB lijnmetwis
KEY(8) ON: ON KEY(8) GOSUB lijnzonderwis
KEY(9) ON: ON KEY(9) GOSUB willekeuroud
KEY(10) ON: ON KEY(10) GOSUB willekeurnieuw
DIM x(100, 103), y(100, 103), z(100, 103), xfz(100), yfz(100), zfz(100)
DIM x2d(200), y2d(200)
SCREEN 12, 0: CLS
xyz = 100
mfz = .1
aantal = 3
scherm = 1
begincord = 1
lijn = 0
willoud = 100
willnieuw = 1
wg = 3 * willoud
```



```

afrond = 0

WINDOW (-wg, wg)-(-wg, -wg)

prog = 1

WHILE prog > 0

CLS

FOR tel = 0 TO aantal - 1

    x(tel, 0) = (RND(1) * 2 * willoud) - willoud: x(tel, 1) = x(tel, 0) + (RND(1) * 2 * willnieuw) - willnieuw
    y(tel, 0) = (RND(1) * 2 * willoud) - willoud: y(tel, 1) = y(tel, 0) + (RND(1) * 2 * willnieuw) - willnieuw
    z(tel, 0) = (RND(1) * 2 * willoud) - willoud: z(tel, 1) = z(tel, 0) + (RND(1) * 2 * willnieuw) - willnieuw

NEXT tel

IF begincord = 1 THEN GOSUB bcord

GOSUB status

prog = 2

WHILE prog > 1

FOR tel1 = 0 TO aantal - 1

    x(tel1, 2) = x(tel1, 1) - x(tel1, 0)
    y(tel1, 2) = y(tel1, 1) - y(tel1, 0)
    z(tel1, 2) = z(tel1, 1) - z(tel1, 0)

    FOR tel2 = tel1 TO aantal - 1

        x(tel1, 3 + tel1) = x(tel2, 1) - x(tel1, 1)
        y(tel1, 3 + tel1) = y(tel2, 1) - y(tel1, 1)
        z(tel1, 3 + tel1) = z(tel2, 1) - z(tel1, 1)

        x(tel2, 3 + tel2) = -x(tel1, 3 + tel1)
        y(tel2, 3 + tel2) = -y(tel1, 3 + tel1)
        z(tel2, 3 + tel2) = -z(tel1, 3 + tel1)

        x(tel1, 3 + aantal + tel1) = ABS(x(tel1, 3 + tel1))
        y(tel1, 3 + aantal + tel1) = ABS(y(tel1, 3 + tel1))
        z(tel1, 3 + aantal + tel1) = ABS(z(tel1, 3 + tel1))

        x(tel2, 3 + aantal + tel2) = ABS(x(tel2, 3 + tel2))
        y(tel2, 3 + aantal + tel2) = ABS(y(tel2, 3 + tel2))
        z(tel2, 3 + aantal + tel2) = ABS(z(tel2, 3 + tel2))

    NEXT tel2

NEXT tel1

FOR tel1 = 0 TO aantal - 1

    xfz(tel1) = 0
    yfz(tel1) = 0
    zfz(tel1) = 0

    FOR tel2 = 0 TO aantal - 1

        IF x(tel1, 3 + aantal + tel2) > 0 THEN xfz(tel1) = xfz(tel1) + x(tel1, 3 + tel2) * mfz / x(tel1, 3 + aantal + tel2)
        IF y(tel1, 3 + aantal + tel2) > 0 THEN yfz(tel1) = yfz(tel1) + y(tel1, 3 + tel2) * mfz / y(tel1, 3 + aantal + tel2)
        IF z(tel1, 3 + aantal + tel2) > 0 THEN zfz(tel1) = zfz(tel1) + z(tel1, 3 + tel2) * mfz / z(tel1, 3 + aantal + tel2)

```

```

NEXT tel2

x(tel1, 0) = x(tel1, 1)

IF afrond = 0 THEN x(tel1, 1) = x(tel1, 0) + x(tel1, 2) + xfz(tel1) ELSE x(tel1, 1) = INT(x(tel1, 0) + x(tel1, 2) + xfz(tel1))

y(tel1, 0) = y(tel1, 1)

IF afrond = 0 THEN y(tel1, 1) = y(tel1, 0) + y(tel1, 2) + yfz(tel1) ELSE y(tel1, 1) = INT(y(tel1, 0) + y(tel1, 2) + yfz(tel1))

z(tel1, 0) = z(tel1, 1)

IF afrond = 0 THEN z(tel1, 1) = z(tel1, 0) + z(tel1, 2) + zfz(tel1) ELSE z(tel1, 1) = INT(z(tel1, 0) + z(tel1, 2) + zfz(tel1))

NEXT tel1

midx = 0

midy = 0

midz = 0

FOR tel = 0 TO aantal - 1

midx = midx + x(tel, 1)

midy = midy + y(tel, 1)

midz = midz + z(tel, 1)

NEXT tel

midx = midx / aantal

midy = midy / aantal

midz = midz / aantal

w2dx = midy - midx * .5

w2dy = midz - midx * .5

IF lijn = 2 THEN GOSUB wislijn:

FOR tel = 0 TO aantal - 1

x2d(tel) = y(tel, 1) - x(tel, 1) * .5

y2d(tel) = z(tel, 1) - x(tel, 1) * .5

NEXT tel

WINDOW (-wg + w2dx, wg + w2dy)-(wg + w2dx, -wg + w2dy)

IF lijn = 0 THEN GOSUB tekenpunt: ELSE GOSUB tekenlijn:

WEND

WEND

andermode:

scherm = scherm + 1

IF scherm > 2 THEN scherm = 0

IF scherm = 0 THEN SCREEN 9, 0: WIDTH 80, 43: COLOR 1, 10

IF scherm = 1 THEN SCREEN 12: WIDTH 80, 60

IF scherm = 2 THEN SCREEN 13

GOSUB status

RETURN

afrondschoonscherm:

IF afrond = 0 THEN afrond = 1 ELSE afrond = 0

GOSUB status

```

```

RETURN

nieuwecoordinaten:

prog = 1

CLS

RETURN

sterktezwaartekracht:

PRINT "Mate van zwaartekracht is:"; mfz

INPUT "Nieuwe mate:"; mfz

GOSUB status

RETURN

nieuwaaantaldeeltjes:

PRINT "Aantal deeltjes is:"; aantal

INPUT "Nieuw aantal:"; aantal

IF aantal < 1 THEN aantal = 1

IF aantal > 50 THEN aantal = 50

CLS

prog = 1

RETURN

windowgrootte:

PRINT "Windowgrootte is:"; wg

INPUT "Nieuwe grootte:"; wg

IF wg < 10 THEN wg = 10

IF wg > 500 THEN wg = 500

GOSUB status:

RETURN

willekeuroud:

PRINT "Randomize oude co rdinaat is:"; willoud

INPUT "Nieuwe randomize factor:"; willoud

IF willoud < 1 THEN willoud = 1

IF willoud > 10000 THEN willoud = 10000

wg = 3 * willoud

CLS

prog = 1

RETURN

willekeurnieuw:

PRINT "Randomize nieuwe co rdinaat is:"; willnieuw

INPUT "Nieuwe randomize factor:"; willnieuw

IF willnieuw < .0000001 THEN willoud = .0000001

IF willnieuw > 1000 THEN willnieuw = 1000

CLS

prog = 1

```

```

RETURN

lijnzonderwis:
IF lijn = 1 THEN lijn = 0 ELSE lijn = 1

CLS

IF lijn = 0 THEN GOSUB status:

RETURN

lijnmetwis:
IF lijn = 2 THEN lijn = 0 ELSE lijn = 2

CLS

IF lijn = 0 THEN GOSUB status:

RETURN

bcord:
beginncord = 0

x(0, 0) = xyz: x(0, 1) = xyz
y(0, 0) = 0: y(0, 1) = -.9
z(0, 0) = 0: z(0, 1) = .9

x(1, 0) = 0: x(1, 1) = .9
y(1, 0) = xyz: y(1, 1) = xyz
z(1, 0) = 0: z(1, 1) = -.9

x(2, 0) = 0: x(2, 1) = -.9
y(2, 0) = 0: y(2, 1) = .9
z(2, 0) = xyz: z(2, 1) = xyz

RETURN

tekenpunt:
FOR tel = 0 TO aantal - 1

    PSET (x2d(tel), y2d(tel)), 7 + tel

NEXT tel

RETURN

tekenlijn:
FOR tel1 = 0 TO aantal - 1

    FOR tel2 = tel1 TO aantal - 1

        LINE (x2d(tel1), y2d(tel1))-(x2d(tel2), y2d(tel2)), 2 + tel1 + tel2

    NEXT tel2

NEXT tel1

RETURN

wislijn:
CLS

RETURN

status:
CLS

IF scherm = 0 THEN PRINT "EGA (16k)"

```

```

IF scherm = 1 THEN PRINT "VGA (16k)"

IF scherm < 2 THEN PRINT "Window-grootte :"; wg

IF scherm < 2 THEN PRINT "Sterkte Fzwaarte:"; mfz

IF scherm < 2 THEN PRINT "Aantal 1db's :"; aantal

IF scherm < 2 THEN PRINT "r_oud :"; willoud

IF scherm < 2 THEN PRINT "r_nieuw :"; willnieuw

IF scherm < 2 THEN PRINT "Afronding c_oud :"; afrond

RETURN

```

2) Program Newton (Borland C)

```

// (C) 1996 G.J. Smit, Nijmegen, Nederland

// This software is published under the GNU General Public License v3.0

// www.dbphysics.com

// The program 'Newton' is a n-body model where simultaneously for the positions of multiple dimensional basics (theoretical infinite curvature)
moving through space time interact with each other according to newtonian laws.

#include "stdio.h"
#include "stdlib.h"
#include "conio.h"
#include "string.h"
#include "float.h"
#include "math.h"
#include "graphics.h"
#include "svga256.h"

FILE *bestand;          // Pointer voor geopend bestand.

char b_naam[12];        // Naam van actief bestand.

char b_test;            // Controle voor bestaan bestand.

char toets;             // Variabele voor ingedrukte toets.

int t1,t2,t3,t4;        // Tellers voor lussen.

int stap,dim,deel;      // Aantal stappen, dimensies en deeltjes.

int spoor;              // Lengte afbeelding in tijd per deeltje.

int prog;              // Programma einde.

int i_temp;            // Tijdelijk opslag integer.

int midd;              // Middelpunt in tekening aan/uit.

int modus;             // Kleur per deel/diepte.

float w_g, w_x, w_y;    // Windowgrootte, x en y co rdinaatgrootte.

float x_max, y_max;     // Aantal pixels op beeldscherm.

float t_frag, grens;    // Tijdfragmentatie en grenswaarde ruimte.

float r_o, r_n;         // Bereik willekeurige beginco rdinaten.

float fzx, fzy, fzz;    // Zwaartekracht per as.

```

```

float g_temp;           // Grenswaarde wisseling.
float f_temp;           // Tijdelijk opslag float.
float midx, midy;       // Middelpunt berekening-variabelen.
float diepte;           // Kleur-diepte variabele.
float huge x3[1851][4]; // Maximaal 250 x3d-co rdinaten.
float huge y3[1851][4]; // Maximaal 250 y3d-co rdinaten.
float huge z3[1851][4]; // Maximaal 250 z3d-co rdinaten.
float huge x2[1851][30]; // Maximaal 30 x2d-co rdinaten per deeltje.
float huge y2[1851][30]; // Maximaal 30 y2d-co rdinaten per deeltje.
int huge DetectSvga256(){ int Vid; Vid=4; return Vid; }
void theorie(void) // Rekenkundig variabele verhoudingen.
{ t1=0; t2=2639;
  installuserdriver("Svga256",DetectSvga256);
  initgraph(&t1,&t2,"");
  setcolor(7);
  for(t1=0; t1<20; t1++)
  { line(x_max/2+200-t1*10, y_max/2, x_max/2, y_max/2+t1*10);
    line(x_max/2-t1*10, y_max/2, x_max/2, y_max/2+200-t1*10);
    line(x_max/2-200+t1*10, y_max/2, x_max/2, y_max/2-t1*10);
    line(x_max/2+t1*10, y_max/2, x_max/2, y_max/2-200+t1*10);
  }
  getch();
  closegraph();
}

void varbestand(void) // Variabele waarden bestand inlezen.
{ bestand=fopen(b_naam,"r");
  if(bestand==NULL) b_test=0;
  else
  { fscanf(bestand, "%d%d%f%f%f%f", &stap, &deel, &r_o, &r_n, &t_frag, &grens);
    if(grens!=0) w_g=1.5*grens;
    b_test=1;
  }
  fclose(bestand);
}

void menuopbeeld(void)
{ clrscr();
  textcolor(10);
  printf("[b]estandsnaam ");
  if(b_test==0) printf("."); else printf("+");
  printf(" : %s\n\n", b_naam);
  printf("[c]o rdinaten\n");

```

```

printf("[w]illekeur      : %f\n", r_o);
printf("[r]ichting      : %f\n\n", r_n);
printf("[t]ijdfragmentatie : %f\n", t_frag);
printf("[g]rens          : %f\n\n", grens);
printf("[s]tappen        : %d\n", stap);
printf("[d]eel           : %d\n\n", deel);
printf("[v]enster        : %f\n", w_g);
printf("[p]rojectie      : %d\n", spoor);
printf("M goedmaken voor 3d: ");
//printf("[m]iddelpunt    : ");
if(midd==0) printf("uit\n"); else printf("aan\n");
printf("[k]leurmodus      : ");
if(modus==0) printf("deel\n\n\n"); else printf("diepte\n\n\n");
printf("[R]ekenen [E]n [T]ekenen [S]toppen\n\n\n");
}

void menuvraag(void)
{ toets=getch();
  if(toets==98) { printf("Nieuwe bestandsnaam? ");
    scanf("%s", &b_naam); }
  if(toets==99) { printf("Invoer co rdinaten, nog programmeren...");
    getch(); }
  if(toets==119) { f_temp=r_o; printf("Maximale willekeur? ");
    scanf("%f", &r_o);
    if(r_o<0 || r_o==0 || r_o>30000) r_o=f_temp; }
  if(toets==114) { f_temp=r_n; printf("Maximale richting? ");
    scanf("%f", &r_n);
    if(r_n<0 || r_o==0 || r_o>2500) r_o=f_temp; }
  if(toets==116) { f_temp=t_frag; printf("Nieuwe tijdfragmentatie? ");
    scanf("%f", &t_frag);
    if(t_frag<0 || t_frag==0 || t_frag>1) t_frag=f_temp; }
  if(toets==103) { f_temp=grens; printf("Grens van ruimte? ");
    scanf("%f", &grens);
    if(grens<0 || grens>32500) grens=f_temp; }
  if(toets==115) { i_temp=stap; printf("Aantal stappen? ");
    scanf("%d", &stap);
    if(stap<1 || stap>32500) stap=i_temp; }
  if(toets==100) { i_temp=deel; printf("Aantal deeltjes? ");
    scanf("%d", &deel);
    if(deel<2 || deel>1850) deel=i_temp; }
  if(toets==118) { f_temp=w_g; printf("Venstergrootte? ");
    scanf("%f", &w_g);

```

```

        if(w_g<0 || w_g==0 || w_g>32500) w_g=f_temp; }

if(toets==112) { i_temp=spoor; printf("Aantal fragmenten? ");

        scanf("%d", &spoor);

        if(spoor<0 || spoor>30) spoor=i_temp; }

if(toets==109) { if(midd==0) midd=1; else midd=0; }

if(toets==107) { if(modus==0) modus=1; else modus=0; }

}

void willekeur(void)

{ for(t1=0;t1<deel;t1++)

    { x3[t1][0]=2*(random(32767)*r_o/32767)-r_o;

      y3[t1][0]=2*(random(32767)*r_o/32767)-r_o;

      z3[t1][0]=2*(random(32767)*r_o/32767)-r_o;

      x3[t1][1]=x3[t1][0]+2*(random(32767)*r_n/32767)-r_n;

      y3[t1][1]=y3[t1][0]+2*(random(32767)*r_n/32767)-r_n;

      z3[t1][1]=z3[t1][0]+2*(random(32767)*r_n/32767)-r_n;

    }

}

void reken(void) // Kaal [R]ekenen, snelste routine.

{ // Bestand voor co rdinaten openen.

    bestand=fopen(b_naam,"w");

    fprintf(bestand, "%d %d %f %f %f %f", stap, deel, r_o, r_n, t_frag, grens);

    for(t1=0;t1<deel;t1++)

        fprintf(bestand, " %f %f %f", x3[t1][1], y3[t1][1], z3[t1][1]);

    // Co rdinaten berekenen, schrijven naar disk en naar tekst-beeldscherm.

    for(t1=0;t1<stap;t1++)

    { for(t2=0;t2<deel;t2++)

        { x3[t2][3]=0;

          y3[t2][3]=0;

          z3[t2][3]=0;

        }

        for(t2=0;t2<deel;t2++)

        { x3[t2][2]=x3[t2][1]-x3[t2][0];

          y3[t2][2]=y3[t2][1]-y3[t2][0];

          z3[t2][2]=z3[t2][1]-z3[t2][0];

          for(t3=t2;t3<deel;t3++)

          { fzx=x3[t3][1]-x3[t2][1];

            fzy=y3[t3][1]-y3[t2][1];

            fzz=z3[t3][1]-z3[t2][1];

            if(fzx!=0) { fzx=1/fzx; x3[t2][3]=x3[t2][3]+fzx;

                        x3[t3][3]=x3[t3][3]-fzx; }

            if(fzy!=0) { fzy=1/fzy; y3[t2][3]=y3[t2][3]+fzy;

```



```

        y3[t3][3]=y3[t3][3]-fzy; }

    if(fzz!=0) { fzz=1/fzz; z3[t2][3]=z3[t2][3]+fzz;

        z3[t3][3]=z3[t3][3]-fzz; }

}

x3[t2][0]=x3[t2][1];
y3[t2][0]=y3[t2][1];
z3[t2][0]=z3[t2][1];

x3[t2][1]=x3[t2][0]+x3[t2][2]+x3[t2][3];
y3[t2][1]=y3[t2][0]+y3[t2][2]+y3[t2][3];
z3[t2][1]=z3[t2][0]+z3[t2][2]+z3[t2][3];

}

for(t2=0;t2<deel;t2++)

printf(bestand, " %f %f %f", x3[t2][1], y3[t2][1], z3[t2][1]);

putchar(13); printf("%d",t1+1);

}

fclose(bestand);

b_test=1;

}

void rekenmetopties(void) // [R]ekenen met grens en/of t_frag aan.

{ // Bestand voor co rdinaten openen.

    bestand=fopen(b_naam,"w");

    printf(bestand, "%d %d %f %f %f", stap, deel, r_o, r_n, t_frag, grens);

    for(t1=0;t1<deel;t1++)

        printf(bestand, " %f %f %f", x3[t1][1], y3[t1][1], z3[t1][1]);

    // Co rdinaten berekenen, schrijven naar disk en naar tekst-beeldscherm.

    for(t1=0;t1<stap;t1++)

    { for(t2=0;t2<deel;t2++)

        { x3[t2][3]=0;

          y3[t2][3]=0;

          z3[t2][3]=0;

        }

        for(t2=0;t2<deel;t2++)

        { x3[t2][2]=x3[t2][1]-x3[t2][0];

          y3[t2][2]=y3[t2][1]-y3[t2][0];

          z3[t2][2]=z3[t2][1]-z3[t2][0];

          for(t3=t2;t3<deel;t3++)

          { fzx=x3[t3][1]-x3[t2][1];

            fzy=y3[t3][1]-y3[t2][1];

            fzz=z3[t3][1]-z3[t2][1];

            if(fzx!=0) { fzx=1/fzx; x3[t2][3]=x3[t2][3]+fzx;

                        x3[t3][3]=x3[t3][3]-fzx; }

```

```

    if(fzy!=0) { fzy=1/fzy; y3[t2][3]=y3[t2][3]+fzy;
                y3[t3][3]=y3[t3][3]-fzy; }
    if(fzz!=0) { fzz=1/fzz; z3[t2][3]=z3[t2][3]+fzz;
                z3[t3][3]=z3[t3][3]-fzz; }
}
// Bewerk co rdinaten als t_frag ongelijk aan 1.
if(t_frag!=1)
{ x3[t2][2]=x3[t2][2]*t_frag; x3[t2][3]=x3[t2][3]*t_frag;
  y3[t2][2]=y3[t2][2]*t_frag; y3[t2][3]=y3[t2][3]*t_frag;
  z3[t2][2]=z3[t2][2]*t_frag; z3[t2][3]=z3[t2][3]*t_frag;
}
// Bepaal de nieuwe co rdinaten.
x3[t2][0]=x3[t2][1];
y3[t2][0]=y3[t2][1];
z3[t2][0]=z3[t2][1];
x3[t2][1]=x3[t2][0]+x3[t2][2]+x3[t2][3];
y3[t2][1]=y3[t2][0]+y3[t2][2]+y3[t2][3];
z3[t2][1]=z3[t2][0]+z3[t2][2]+z3[t2][3];
// Test grensoverschrijding.
if(grens>0)
{ if(x3[t2][1]<-grens | x3[t2][1]>grens)
  { g_temp=x3[t2][1]; x3[t2][1]=-x3[t2][0]; x3[t2][0]=-g_temp; }
  if(y3[t2][1]<-grens | y3[t2][1]>grens)
  { g_temp=y3[t2][1]; y3[t2][1]=-y3[t2][0]; y3[t2][0]=-g_temp; }
  if(z3[t2][1]<-grens | z3[t2][1]>grens)
  { g_temp=z3[t2][1]; z3[t2][1]=-z3[t2][0]; z3[t2][0]=-g_temp; }
}
}
for(t2=0;t2<deel;t2++)
fprintf(bestand, " %f %f %f", x3[t2][1], y3[t2][1], z3[t2][1]);
putchar(13); printf("%d",t1+1);
}
fclose(bestand);
b_test=1;
}
void rekenenteken(void) // [E]n.
{ t1=0; t2=2639;
  installuserdriver("Svga256",DetectSvga256);
  initgraph(&t1,&t2,"");
  w_x=(x_max+1)/(w_g*2); w_y=(y_max+1)/(w_g*2);
  printf("    |%d| %d| %f| %f| %f| %f| %f| %d| %s",

```

```

    stap, deel, r_o, r_n, t_frag, grens, w_g, spoor, b_naam);

gotoxy(0,0);

midd=0; spoor=0; modus=0;

// Bestand voor co rdinaten openen.

bestand=fopen(b_naam,"w");

fprintf(bestand, "%d %d %f %f %f %f", stap, deel, r_o, r_n, t_frag, grens);

for(t1=0;t1<deel;t1++)

fprintf(bestand, " %f %f %f", x3[t1][1], y3[t1][1], z3[t1][1]);

// Co rdinaten berekenen, schrijven naar disk en naar grafisch beeldscherm.

for(t1=0;t1<stap;t1++)

{ for(t2=0;t2<deel;t2++)

{ x3[t2][3]=0;

  y3[t2][3]=0;

  z3[t2][3]=0;

}

for(t2=0;t2<deel;t2++)

{ x3[t2][2]=x3[t2][1]-x3[t2][0];

  y3[t2][2]=y3[t2][1]-y3[t2][0];

  z3[t2][2]=z3[t2][1]-z3[t2][0];

  for(t3=t2;t3<deel;t3++)

  { fzx=x3[t3][1]-x3[t2][1];

    fzy=y3[t3][1]-y3[t2][1];

    fzz=z3[t3][1]-z3[t2][1];

    if(fzx!=0) { fzx=1/fzx; x3[t2][3]=x3[t2][3]+fzx;

                x3[t3][3]=x3[t3][3]-fzx; }

    if(fzy!=0) { fzy=1/fzy; y3[t2][3]=y3[t2][3]+fzy;

                y3[t3][3]=y3[t3][3]-fzy; }

    if(fzz!=0) { fzz=1/fzz; z3[t2][3]=z3[t2][3]+fzz;

                z3[t3][3]=z3[t3][3]-fzz; }

  }

// Bewerk co rdinaten als t_frag ongelijk aan 1.

if(t_frag!=1)

{ x3[t2][2]=x3[t2][2]*t_frag; x3[t2][3]=x3[t2][3]*t_frag;

  y3[t2][2]=y3[t2][2]*t_frag; y3[t2][3]=y3[t2][3]*t_frag;

  z3[t2][2]=z3[t2][2]*t_frag; z3[t2][3]=z3[t2][3]*t_frag;

}

// Bepaal de nieuwe co rdinaten.

x3[t2][0]=x3[t2][1];

y3[t2][0]=y3[t2][1];

z3[t2][0]=z3[t2][1];

x3[t2][1]=x3[t2][0]+x3[t2][2]+x3[t2][3];

```

```

y3[t2][1]=y3[t2][0]+y3[t2][2]+y3[t2][3];
z3[t2][1]=z3[t2][0]+z3[t2][2]+z3[t2][3];

// Test grensoverschrijding.
if(grens>0)
{
if(x3[t2][1]<-grens | x3[t2][1]>grens)
{
g_temp=x3[t2][1]; x3[t2][1]=-x3[t2][0]; x3[t2][0]=-g_temp; }
if(y3[t2][1]<-grens | y3[t2][1]>grens)
{
g_temp=y3[t2][1]; y3[t2][1]=-y3[t2][0]; y3[t2][0]=-g_temp; }
if(z3[t2][1]<-grens | z3[t2][1]>grens)
{
g_temp=z3[t2][1]; z3[t2][1]=-z3[t2][0]; z3[t2][0]=-g_temp; }
}
}

for(t2=0;t2<deel;t2++)

fprintf(bestand, " %f %f %f", x3[t2][1], y3[t2][1], z3[t2][1]);

for(t2=0;t2<deel;t2++)

{
x2[t2][0]=y3[t2][1]-x3[t2][1];
y2[t2][0]=-z3[t2][1]+x3[t2][1]/2;
x2[t2][0]=x_max/2-w_x*x2[t2][0];
y2[t2][0]=y_max/2-w_y*y2[t2][0];
putpixel(x2[t2][0],y2[t2][0],2+t2);
}

putchar(13); printf("%d",t1+1);
}

putchar(13); printf("Klaar");
fclose(bestand);

b_test=1;

getch();

closegraph();
}

void geendisk(void) // [A]lleen rekenen en tekenen.
{
t1=0; t2=2639;

installuserdriver("Svga256",DetectSvga256);

initgraph(&t1,&t2,"");

w_x=(x_max+1)/(w_g*2); w_y=(y_max+1)/(w_g*2);

if(modus!=0) diepte=255/(w_g*2);

printf("   |%d| %d| %f| %f| %f| %f| %f| %d| XXXXXX",
      stap, deel, r_o, r_n, t_frag, grens, w_g, spoor);

gotoxy(0,0);

// Co rdinaten berekenen, schrijven naar grafisch beeldscherm.

for(t1=0;t1<stap;t1++)

{
for(t2=0;t2<deel;t2++)

```

```

{ x3[t2][3]=0;
  y3[t2][3]=0;
  z3[t2][3]=0;
}
putchar(13);printf("%d", t1);
if(midd>0) { midx=0; midy=0; }
for(t2=0;t2<deel;t2++)
{ x3[t2][2]=x3[t2][1]-x3[t2][0];
  y3[t2][2]=y3[t2][1]-y3[t2][0];
  z3[t2][2]=z3[t2][1]-z3[t2][0];
  for(t3=t2;t3<deel;t3++)
  { fzx=x3[t3][1]-x3[t2][1];
    fzy=y3[t3][1]-y3[t2][1];
    fzz=z3[t3][1]-z3[t2][1];
    if(fzx!=0) { fzx=1/fzx; x3[t2][3]=x3[t2][3]+fzx;
                x3[t3][3]=x3[t3][3]-fzx; }
    if(fzy!=0) { fzy=1/fzy; y3[t2][3]=y3[t2][3]+fzy;
                y3[t3][3]=y3[t3][3]-fzy; }
    if(fzz!=0) { fzz=1/fzz; z3[t2][3]=z3[t2][3]+fzz;
                z3[t3][3]=z3[t3][3]-fzz; }
  }
  // Bewerk co rdinaten als t_frag ongelijk aan 1.
  if(t_frag!=1)
  { x3[t2][2]=x3[t2][2]*t_frag; x3[t2][3]=x3[t2][3]*t_frag;
    y3[t2][2]=y3[t2][2]*t_frag; y3[t2][3]=y3[t2][3]*t_frag;
    z3[t2][2]=z3[t2][2]*t_frag; z3[t2][3]=z3[t2][3]*t_frag;
  }
  // Bepaal de nieuwe co rdinaten.
  x3[t2][0]=x3[t2][1];
  y3[t2][0]=y3[t2][1];
  z3[t2][0]=z3[t2][1];
  x3[t2][1]=x3[t2][0]+x3[t2][2]+x3[t2][3];
  y3[t2][1]=y3[t2][0]+y3[t2][2]+y3[t2][3];
  z3[t2][1]=z3[t2][0]+z3[t2][2]+z3[t2][3];
  // Test grensoverschrijding.
  if(grens>0)
  { if(x3[t2][1]<-grens || x3[t2][1]>grens)
    { g_temp=x3[t2][1]; x3[t2][1]=-x3[t2][0]; x3[t2][0]=-g_temp; }
    if(y3[t2][1]<-grens || y3[t2][1]>grens)
    { g_temp=y3[t2][1]; y3[t2][1]=-y3[t2][0]; y3[t2][0]=-g_temp; }
    if(z3[t2][1]<-grens || z3[t2][1]>grens)

```

```

    { g_temp=z3[t2][1]; z3[t2][1]=-z3[t2][0]; z3[t2][0]=-g_temp; }
}
}
for(t2=0;t2<deel;t2++)
{ x2[t2][0]=y3[t2][1]-x3[t2][1];
  y2[t2][0]=-z3[t2][1]+x3[t2][1]/2;
  if(midd>0)
  { midx=midx+x2[t2][0];
    midy=midy+y2[t2][0];
  }
}
if(midd>0) { midx=midx/deel; midy=midy/deel; }
for(t2=0;t2<deel;t2++)
{ if(midd>0)
  { x2[t2][0]=x_max/2+midx-w_x*x2[t2][0];
    y2[t2][0]=y_max/2+midy-w_y*y2[t2][0];
  }
  else
  { x2[t2][0]=x_max/2-w_x*x2[t2][0];
    y2[t2][0]=y_max/2-w_y*y2[t2][0];
  }
  if(modus==0) putpixel(x2[t2][0],y2[t2][0],2+t2);
  else putpixel(x2[t2][0],y2[t2][0],1+(x3[t2][1]+w_g)*diepte);
  if(spoor>0)
  { if(t1<spoor-1)
    { x2[t2][spoor-1-t1]=x2[t2][0]; y2[t2][spoor-1-t1]=y2[t2][0]; }
    else
    { putpixel(x2[t2][spoor-1],y2[t2][spoor-1],0);
      for(t3=spoor-1;t3>0;t3--)
      { x2[t2][t3]=x2[t2][t3-1]; y2[t2][t3]=y2[t2][t3-1]; }
    }
  }
}
putchar(13); printf("%d",t1+1);
}
putchar(13); printf("Klaar");
getch();
closegraph();
}

void teken(void)

```

```

{ t1=0; t2=2639;

installuserdriver("Svg256",DetectSvg256);

initgraph(&t1,&t2,"");

// Berekenen hoeveelheid pixels per nheid.

w_x=(x_max+1)/(w_g*2); w_y=(y_max+1)/(w_g*2);

bestand=fopen(b_naam,"r");

fscanf(bestand, "%d%d%f%f%f%", &stap, &deel, &r_o, &r_n, &t_frag, &grens);

printf("  |%d| %d| %f| %f| %f| %f| %f| %d| %s",

        stap, deel, r_o, r_n, t_frag, grens, w_g, spoor, b_naam);

if(modus!=0) diepte=255/(w_g*2);

for(t1=0;t1<stap+1;t1++)

{ putchar(13);printf("%d", t1);

if(midd>0) { midx=0; midy=0; }

for(t2=0;t2<deel;t2++)

fscanf(bestand, "%f%f%f%", &x3[t2][1], &y3[t2][1], &z3[t2][1]);

for(t2=0;t2<deel;t2++)

{ x2[t2][0]=y3[t2][1]-x3[t2][1];

y2[t2][0]=-z3[t2][1]+x3[t2][1]/2;

if(midd>0)

{ midx=midx+x2[t2][0];

midy=midy+y2[t2][0];

}

}

if(midd>0) { midx=midx/deel; midy=midy/deel; }

for(t2=0;t2<deel;t2++)

{ if(midd>0)

{ x2[t2][0]=x_max/2+midx-w_x*x2[t2][0];

y2[t2][0]=y_max/2+midy-w_y*y2[t2][0];

}

else

{ x2[t2][0]=x_max/2-w_x*x2[t2][0];

y2[t2][0]=y_max/2-w_y*y2[t2][0];

}

if(modus==0) putpixel(x2[t2][0],y2[t2][0],2+t2);

else putpixel(x2[t2][0],y2[t2][0],1+(x3[t2][1]+w_g)*diepte);

if(spoor>0)

{ if(t1<spoor-1)

{ x2[t2][spoor-1-t1]=x2[t2][0]; y2[t2][spoor-1-t1]=y2[t2][0]; }

else

{ putpixel(x2[t2][spoor-1],y2[t2][spoor-1],0);

for(t3=spoor-1;t3>0;t3--)

```

```

        { x2[t2][t3]=x2[t2][t3-1]; y2[t2][t3]=y2[t2][t3-1]; }
    }
}
}
putchar(13); printf("Klaar");
fclose(bestand);
getch();
closegraph();
}

void main(void)
{ // Beginwaarden zetten.

    stap=250; deel=3; r_o=100; r_n=.0001; t_frag=1; grens=0; w_g=500; spoor=0;

    midd=0; spoor=0; modus=0;

    clrscr(); textcolor(10);

    // Grafische modus bepalen.

    t1=0; t2=2639;

    installuserdriver("Svga256",DetectSvga256);

    initgraph(&t1,&t2,"");

    x_max=getmaxx(); y_max=getmaxy();

    closegraph();

    // Test of standaard bestand bestaat.

    strcpy(b_naam, "bestand.xyz");

    b_test=0;

    varbestand();

    // Begin programma-lus.

    prog=1;

    do

    { menuopbeeld();

        toets=0;

        menuvraag();

        // [b]estandsnaam.

        if(toets==98) varbestand();

        // [R]eken.

        if(toets==82)

        { if(b_test==1)

            { printf("Bestand %s overschrijven? [j/n] ", b_naam);

                i_temp=getch();

                putchar(13); printf("                                ");

                putchar(13);

                if(i_temp==106) b_test=0;

```



```

    }
    if(b_test==0)
    { willekeur();
      if(grens>0 || t_frag!=1) rekenmetopties(); else reken();
    }
  }
  if(toets==69)
  { if(b_test==1)
    { printf("Bestand %s overschrijven? [j/n] ", b_naam);
      i_temp=getch();
      putchar(13); printf("                ");
      putchar(13);
      if(i_temp==106) b_test=0;
    }
    if(b_test==0)
    { willekeur();
      rekenenteken();
    }
  }
  // [T]eken.
  if(toets==84) teken();
  if(toets==65) { willekeur(); geendisk(); }
  if(toets==81) theorie();
  if(toets==63) { printf("Bedacht en geschreven door G.J.Smit.");
                 getch(); }
  if(toets==83) prog=0;
} while(prog>0);
}

```

3) Program Einstein (Borland C)

```
// (C) 1996 G.J. Smit, Nijmegen, Nederland
```

```
// This software is published under the GNU General Public License v3.0
```

```
// www.dbphysics.com
```

```
// The program 'Einstein' photographs (plots) a piece of einsteinian space time where individual and multiple dimensional basics can be seen, showing the deformation of space time as seen for an outside observer.
```

```
#include "conio.h"
```

```
#include "graphics.h"
```

```
#include "math.h"
```

```
#include "process.h"
```

```

#include "stdio.h"

#include "stdlib.h"

#include "string.h"

FILE *vkini;           // Actieve rekenvariabelen.

FILE *vkxyz;           // Krommingssterkte en virtuele 3D-coördinaten.

FILE *vkfilm;          // Film krommingsverloop.

char fiotest;          // Menu +/- controle op bestaande bestanden.

char prog;             // Stuur programmaverloop.

float xd[24],yd[24],zd[24]; // Coördinaten van maximaal 24 1db's.

char deel,dtel,ctel;    // Actieve hoeveelheid 1db's en teller daarvoor en teller voor invoer coördinaten.

float bereik,stap;      // Formaat en resolutie van berekende ruimte-kubus.

float kromming,afstand; // Sterkte en spreiding van de zichtbare kromming.

char ruimte;           // Al of niet afbeelden als gekromde ruimte.

float schaal;           // Grootte van afbeelding op scherm.

char kl_modus;          // Kleurenpalette/kleurmodus.

float dummy,begin,eind; // Waarden voor film.

float frag;            // Voor film krommingsverloop.

char film;             // Bepaalt film aan/uit in tekenfunctie.

int ftel,fx,fy;         // Besturing film.

unsigned far fk;        // Besturing film.

char toets;            // Test op toetsaanslag in menu.

float x,y,z;           // Actieve rekencoördinaten.

float afx,afy,afz,afs,krm; // Berekenen krommingssterkten.

float xtot,ytot,ztot;   // Berekenen visuele coördinaten.

float ktot;            // Berekenen totale krommingssterkte per coördinaat.

int v1,v2;             // Instellen video-mode.

float x2,y2;           // 2D coördinaten voor het beeldscherm.

float kleur,midx,midy;  // Kleur van te tekenen pixel + relocatie.

float afst,kl_w;        // Voor tekenen in kl_modus=2.

char c_invoer;         // Voor invoer coördinaten.

char bnaam[13],tnaam[13]; // Voor variabele bestandsnaam

int huge DetectSvga256() { int vid; vid=4; return vid; }

void kleur_mod(void)

{ v1=0;v2=2341;

  installuserdriver("Svga256",DetectSvga256);

  initgraph(&v1,&v2,"");

  midx=getmaxx()/2;midy=getmaxy()/2;

  if(kl_modus==0) { for (dtel=0;dtel<63;dtel++) setrgbpalette(32+dtel,dtel,dtel,dtel); }

  if(kl_modus==1)

  { for (dtel=0;dtel<64;dtel++) setrgbpalette(128+dtel,63-dtel,dtel,0);

    for (dtel=0;dtel<64;dtel++) setrgbpalette(192+dtel,0,63-dtel,dtel);
  }
}

```

```

}

if(kl_modus==2)
{
for (dtel=0;dtel<32;dtel++) setrgbpalette(32+dtel,2*dtel,2*dtel,2*dtel);

for (dtel=0;dtel<32;dtel++) setrgbpalette(64+dtel,2*dtel,0,0);

for (dtel=0;dtel<32;dtel++) setrgbpalette(96+dtel,0,2*dtel,0);

for (dtel=0;dtel<32;dtel++) setrgbpalette(128+dtel,0,0,2*dtel);

for (dtel=0;dtel<32;dtel++) setrgbpalette(160+dtel,2*dtel,2*dtel,0);

for (dtel=0;dtel<32;dtel++) setrgbpalette(192+dtel,0,2*dtel,2*dtel);

for (dtel=0;dtel<32;dtel++) setrgbpalette(224+dtel,2*dtel,0,2*dtel);

}

setfillstyle(1,0);

}

void reken(void)
{
_setcursortype(_NOCURSOR);

strcpy(tnaam,bnaam); strcat(tnaam,".ini");

vkini=fopen(tnaam,"wb");

fprintf(vkini,"%d %f %f",deel,bereik,stap);

for (dtel=0;dtel<deel;dtel++) fprintf(vkini,"%f %f %f",xd[dtel],yd[dtel],zd[dtel]);

fclose(vkini);

// Bestand openen voor krommingssterkte en visuele 3D coördinaten.

strcpy(tnaam,bnaam); strcat(tnaam,".xyz");

vkxyz=fopen(tnaam,"wb");

// Berekenen krommingssterkten per coördinaat per deeltje in kubus.

for (x=-bereik;x<bereik;x+=stap)

{ gotoxy(14,19); printf("  %5.3f procent",x/(2*bereik)*100+50);

for (y=-bereik;y<bereik;y+=stap)

{ for (z=-bereik;z<bereik;z+=stap)

{ ktot=0; xtot=0; ytot=0; ztot=0;

for (dtel=0;dtel<deel;dtel++)

{ afx=(x-xd[dtel])*(x-xd[dtel]); // Afstand per x,y,z as.

afy=(y-yd[dtel])*(y-yd[dtel]);

afz=(z-zd[dtel])*(z-zd[dtel]);

afs=sqrt(afx+afy+afz); // Afstand coördinaat tot deeltje.

if(afs!=0) krm=1/(afs*afs); else krm=1000000; // Krommingssterkte bepalen.

// Bepalen coördinaten voor representatie van visuele ruimte door krommingssterkte.

ktot+=krm;

xtot+=(x-xd[dtel])/krm;

ytot+=(y-yd[dtel])/krm;

ztot+=(z-zd[dtel])/krm;

}

}

fprintf(vkxyz,"%f %f %f %f",ktot,xtot,ytot,ztot);

```

```

    }
}
if(kbhit()!=0)
{ if(getch()==27) x=bereik;
}
}
fclose(vkxyz);
if(fioteest==0 || fioteest==1) fioteest=1; else fioteest=3;
if(toets==27)
{ unlink(tnaam);
  if(fioteest==3) fioteest=2; else fioteest=0;
}
toets=32;
}
void teken(void)
{ if(film==0)
{ strcpy(tnaam,bnaam); strcat(tnaam,".ini");
  vkini=fopen(tnaam,"rb");
  fscanf(vkini,"%d%f%f",&deel,&bereik,&stap);
  for(dtel=0;dtel<deel;dtel++) fscanf(vkini,"%f%f%f",xd[dtel],yd[dtel],zd[dtel]);
  fclose(vkini);
  printf("Deel:%d Bereik:%f Stap:%f Kromming:%f Afstand:%f Ruimte:%d Kleur:%d Schaal:%f",
    deel,bereik,stap,kromming,afstand,ruimte,kl_modus,schaal);
}
else
{ bar(0,0,1023,767);
  gotoxy(1,1);
  printf("FC : Begin:%f Eind:%f Fragment:%f Kromming:%f Afstand:%f Ruimte:%d Kleur:%d Schaal:%f",
    begin,eind,frag,kromming,afstand,ruimte,kl_modus,schaal);
}
setcolor(7);line(0,18,1023,18);setcolor(10);
strcpy(tnaam,bnaam); strcat(tnaam,".xyz");
vkxyz=fopen(tnaam,"rb");
for(x=-bereik;x<bereik;x+=stap)
{ line((x/(2*bereik)*100+50)*10.24,18,((x+stap)/(2*bereik)*100+50)*10.24,18);
  for(y=-bereik;y<bereik;y+=stap)
  { for(z=-bereik;z<bereik;z+=stap)
    { fscanf(vkxyz,"%f %f %f %f",&ktot,&xtot,&ytot,&ztot);
      if(ktot>kromming-afstand&&ktot<kromming+afstand)
      { if(ruimte==0) { xtot=x; ytot=y; ztot=z; }
        x2=(ytot-.5*xtot)*schaal;

```

```

y2=(ztot-.5*xtot)*schaal;

if(kl_modus==0) kleur=(x+bereik)/(2*bereik)*62+1;

if(kl_modus==1)
{ if(ktot>kromming&&ktot<(kromming+afstand)) kleur=160+64*(ktot-kromming)/afstand;
  else kleur=160+64*(ktot-kromming)/afstand;
}

if(kl_modus==2)
{ afst=1000000;
  kl_w=32;
  for(dtel=0;dtel<deel;dtel++)
  { afx=(x-xd[dtel])*(x-xd[dtel]);
    afy=(y-yd[dtel])*(y-yd[dtel]);
    afz=(z-zd[dtel])*(z-zd[dtel]);
    afs=sqrt(afx+afy+afz);
    if(afst>afs)
    { afst=afs;
      kl_w=32*dtel;
    }
  }
  kleur=(x+bereik)/(2*bereik)*30+1+kl_w;
}

putpixel(midx+x2,midy-y2,32+kleur);
}

if(kbhit()!=0)
{ if(getch()==27)
{ if(ruimte==0)
  printf("\n\nOnderbroken k:%f x:%f y:%f z:%f x2:%f y2:%f",ktot,x,y,z,x2,y2);
  else
  printf("\n\nOnderbroken k:%f x:%f y:%f z:%f x2:%f y2:%f",ktot,xtot,ytot,ztot,x2,y2);
  x=bereik; y=bereik; z=bereik;
  if(film!=0) // Als film creëren dan einde beeld-lus.
  { kromming=eind;
    film=0;
  }
}
}
}
}
}

fclose(vkxyz);

if(film==0)

```

```

{ getch();
  toets=32;
}
else
{ for(fy=19;fy<768;fy++)
  { for(fx=0;fx<1024;fx++)
    { fk=getpixel(fx,fy);
      if(fk!=0) fprintf(vkfilm,"%d %d %u ",fx,fy,fk);
    }
  }
}
}

void animatie(void)
{ putchar(13);
  printf("KF : Begin:%f Eind:%f Fragment:%f Ruimte:%d Kleur:%d Schaal:%f",
        begin,eind,frag,ruimte,kl_modus,schaal);

  vkfilm=fopen("vkfilm.xyz","rb");
  do
  { fscanf(vkfilm,"%d",&ftel);
    if(ftel<0)
    { if(ftel!=-1)
      { if(ftel==-1000) printf("          Laatste beeld");
        toets=getch();
        if(toets!=27) bar(0,19,1023,767);
        else toets=32;
      }
      fscanf(vkfilm,"%d",&fx);
    }
    else fx=ftel;
    fscanf(vkfilm,"%d",&fy);
    fscanf(vkfilm,"%u",&fk);
    putpixel(fx,fy,fk);
  } while(ftel>-999);
  fclose(vkfilm);
}

void cord_in(void) // Nu 24 deeltjes in te voeren
{
  for(dtel=0;dtel<deel;dtel++)
  { clrscr();
    printf("Bestandsnaa[m] : %s\n\n",bnaam);
  }
}

```

```

printf("[T]ekenen ");
if(fioteest==1 || fioteest==3) printf("+"); else printf("-");
printf(" [R]ekenen\n");
printf("[F]ilm ");
if(fioteest==2 || fioteest==3) printf("+"); else printf("-");
printf(" [C]reëren\n");
printf("[w]illekeur [i]nvoeren\n");
printf("[d]eel : %d\n",deel);
printf("[b]ereik : %f\n",bereik);
printf("[s]tap : %f\n",stap);
printf("[k]romming : %f\n",kromming);
printf("[a]fstand : %f\n",afstand);
printf("[r]uimte : ");
if(ruimte==0) printf("lineair\n"); else printf("gekromd\n");
printf("[s]chaal : %f\n",schaal);
printf("[k]leur : ");
if(kl_modus==0) printf("3D grijswaarden\n");
if(kl_modus==1) printf("2D krommingssterkte\n");
if(kl_modus==2) printf("3D per deeltje\n");
printf("[b]egin : %f\n",begin);
printf("[e]ind : %f\n",eind);
printf("[f]ragment : %f\n",frag);
for(ctel=0;ctel<dtel;ctel++)
{ gotoxy(38,1+ctel);
printf("%d : %f %f %f",ctel,xd[ctel],yd[ctel],zd[ctel]);
}
gotoxy(1,19); printf("Deel : %d\n",dtel);
printf("x > "); scanf("%f",&xd[dtel]);
printf("y > "); scanf("%f",&yd[dtel]);
printf("z > "); scanf("%f",&zd[dtel]);
}
toets=32;
}

void main(void)
{ // Bepalen van beginwaarden voor actieve rekenvariabelen.
strcpy(bnaam,"vkveld");
strcpy(tnaam,bnaam); strcat(tnaam,".ini");
vkini=fopen(tnaam,"rb");
if(vkini==NULL)
{ fioteest=0;
deel=7; bereik=10; stap=.25;

```

```

for (dtel=0;dtel<deel;dtel++)
{
    xd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;
    yd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;
    zd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;
}
}
else
{
    fiotest=1;

    fscanf(vkini,"%d%f%f",&deel,&bereik,&stap);

    for (dtel=0;dtel<deel;dtel++)
    {
        fscanf(vkini,"%f%f%f",&xd[dtel],&yd[dtel],&zd[dtel]);
    }
}

fclose(vkini);

// Testen of vkveld.xyz bestaat.

strcpy(tnaam,bnaam); strcat(tnaam,".xyz");

vkxyz=fopen(tnaam,"rb");

if(vkxyz==NULL) fiotest=0;

fclose(vkxyz);

// Bepalen van beginwaarden voor actieve tekenvariabelen.

kromming=1; afstand=.25; schaal=10; ruimte=0; kl_modus=0;

// Bepalen van beginwaarden voor actieve filmvariabelen.

vkini=fopen("vkfilm.ini","rb");

if(vkini==NULL)

{
    begin=.5; eind=1.5; frag=.25;
}

else

{
    if(fiotest==0) fiotest=2; else fiotest=3;

    fscanf(vkini,"%f%f%f%f",&begin,&eind,&frag,&kl_modus);
}

fclose(vkini);

// Testen of vkfilm.xyz bestaat.

vkfilm=fopen("vkfilm.xyz","rb");

if(vkfilm==NULL) { if(fiotest==1 || fiotest==3) fiotest=1; else fiotest=0; }

fclose(vkfilm);

// Menu -> begin programma-lus.

prog=1; do

{
    // Menu op het scherm.

    _setcursortype(_NOCURSOR);

    clrscr();

    printf("Bestandsnaa[m] : %s\n",bnaam);

```



```

printf("[T]ekenen ");
if(fioteest==1 || fioteest==3) printf("+"); else printf("-");
printf(" [R]ekenen\n");
printf("[F]ilm ");
if(fioteest==2 || fioteest==3) printf("+"); else printf("-");
printf(" [C]reëren\n");
printf("[w]illekeur [i]nvoeren\n");
printf("[d]eel : %d\n",deel);
printf("[b]ereik : %f\n",bereik);
printf("[s]tap : %f\n",stap);
printf("[k]romming : %f\n",kromming);
printf("[a]fstand : %f\n",afstand);
printf("[r]uimte : ");
if(ruimte==0) printf("lineair\n"); else printf("gekromd\n");
printf("[s]haal : %f\n",schaal);
printf("[k]leur : ");
if(kl_modus==0) printf("3D grijswaarden\n");
if(kl_modus==1) printf("2D krommingssterkte\n");
if(kl_modus==2) printf("3D per deeltje\n");
printf("[b]egin : %f\n",begin);
printf("[e]ind : %f\n",eind);
printf("[f]ragment : %f\n",frag);
// Bestandsnaam afdrukken
// Coördinaten op het scherm.
for(dtel=0;dtel<deel;dtel++)
{
    gotoxy(38,1+dtel);
    printf("%d : %f %f %f",dtel,xd[dtel],yd[dtel],zd[dtel]);
}
// Keuze voor functie en afhandeling daarvan.
toets=getch(); gotoxy(1,19); _setcursortype(_NORMALCURSOR);
if(toets==27) prog=0;
if(toets==100)
{
    printf("Aantal 1db's ? "); scanf("%d",&deel);
    if(deel<1) deel=1; if(deel>24) deel=24;
    if(kl_modus==2) kl_modus=0;
}
if(toets==98)
{
    printf("Maximale coördinaten ? "); scanf("%f",&bereik);
    if(bereik==0) bereik=10; if(bereik<0) bereik=-bereik;
}
if(toets==115)

```

```

{ gotoxy(1,19); printf("Resolutie ? "); scanf("%f",&stap);

    if(stap==0) stap=.25; if(stap<0) stap=-stap;

}

if(toets==119)

{ for(dtel=0;dtel<deel;dtel++)

    { xd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;

      yd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;

      zd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;

    }

}

if(toets==105) cord_in();

if(toets==107)

{ printf("Zichtbare kromming ? "); scanf("%f",&kromming);

    if(kromming==0) kromming=1; if(kromming<0) kromming=-kromming;

}

if(toets==97)

{ printf("Afstand tot zichtbare kromming ? "); scanf("%f",&afstand);

    if(afstand==0) afstand=.25; if(afstand<0) afstand=-afstand;

}

if(toets==114)

{ if(ruimte==0) ruimte=1; else ruimte=0;

}

if(toets==99)

{ printf("Grootte (op 1024x768) ? "); scanf("%f",&schaal);

    if(schaal==0) schaal=10; if(schaal<0) schaal=-schaal;

}

if(toets==108)

{ if(kl_modus==0) kl_modus=1;

    else

    { if(kl_modus==1)

        { if(deel<8) kl_modus=2; else kl_modus=0; }

        else

        { if(kl_modus==2) kl_modus=0; }

    }

}

if(toets==101)

{ printf("Beginkromming ? "); scanf("%f",&begin);

    if(begin<0) begin=-begin;

}

if(toets==110)

{ printf("Eindkromming ? "); scanf("%f",&eind);

```

```

    if(eind<0) eind=-eind;
}
if(toets==102)
{ printf("Fragmentatie ? "); scanf("%f",&frag);
}
if(toets==109)
{ printf("Nieuwe bestandsnaam ? "); scanf("%s",&bnaam);
// Testen of bnaam.ini bestaat.
strcpy(tnaam,bnaam); strcat(tnaam,".ini");
vkini=fopen(tnaam,"rb");
if(vkini==NULL)
{ fiotest=0;
// deel=7; bereik=10; stap=.25;
// for(dtel=0;dtel<deel;dtel++)
// { xd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;
// yd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;
// zd[dtel]=((random(32767)*bereik)/32767-bereik/2)*2;
// }
}
else
{ fiotest=1;
fscanf(vkini,"%d%f%f",&deel,&bereik,&stap);
for(dtel=0;dtel<deel;dtel++)
{ fscanf(vkini,"%f%f%f",&xd[dtel],&yd[dtel],&zd[dtel]);
}
}
fclose(vkini);
// Testen of bnaam.xyz bestaat.
strcpy(tnaam,bnaam); strcat(tnaam,".xyz");
vkxyz=fopen(tnaam,"rb");
if(vkxyz==NULL) fiotest=0;
fclose(vkxyz);
}
if(toets==82)
{ if(fiotest==1 || fiotest==3)
{ printf("Bestand overschrijven (j/n) ? ");
toets=getch();
if(toets==106)
{ putchar(13); printf("
");
putchar(13); printf("Rekenen");
reken(); toets=32;

```

```

    }
}
else
{ putchar(13); printf("Rekenen");
  reken();
}
}
if(toets==84)
{ if(fioteest==0 | fioteest==2)
  { printf("Bestand bestaat niet ! "); getch();
  }
  else
  { film=0;
    kleur_mod();
    teken();
    closegraph();
  }
}
if(toets==67) // Film creëren.
{ if(fioteest==3) // Test of vkfilm.xyz bestaat.
  { printf("Bestand overschrijven (j/n) ? ");
    toets=getch();
    if(toets==106) fioteest=1;
  }
  if(fioteest==1)
  { // Kijk naar variabelen voor bereik & stap.
    vkini=fopen("vkveld.ini","rb");
    fscanf(vkini,"%d%f%f",&deel,&bereik,&stap);
    for (dtel=0;dtel<deel;dtel++)
    { fscanf(vkini,"%f%f%f",&xd[dtel],&y[dtel],&z[dtel]);
    }
    fclose(vkini);
    // Begin beeld-lus.
    kleur_mod();
    vkini=fopen("vkfilm.ini","wb");
    fprintf(vkini,"%f %f %f %d",begin,eind,frag,kl_modus);
    fclose(vkini);
    vkfilm=fopen("vkfilm.xyz","wb");
    film=1; ftel=1;
    for (kromming=begin;kromming<eind;kromming+=frag)
    { fprintf(vkfilm,"%d ",ftel);

```

```

    teken();
    ftel++;
}
fprintf(vkfilm,"-1000");
fclose(vkfilm);
fiotest=3;
if(film==0)
{ fiotest=1;
  unlink("vkfilm.xyz");
}
closegraph();
}
if(fiotest==0 || fiotest==2)
{ printf("Bestand bestaat niet ! "); getch();
}
toets=32;
}
if(toets==70)
{ if(fiotest==3)
{ vkini=fopen("vkveld.ini","rb");
  fscanf(vkini,"%d%f%",&deel,&bereik,&stap);
  for (dtel=0;dtel<deel;dtel++)
  { fscanf(vkini,"%f%f%",&xd[dtel],&y[dtel],&z[dtel]);
  }
  fclose(vkini);
  kleur_mod();
  animatie();
  closegraph();
}
if(fiotest==2)
{ kleur_mod();
  animatie();
  closegraph();
}
if(fiotest<2)
{ printf("Bestand bestaat niet ! "); getch();
}
toets=32;
}
if(toets==63)
{ for (dtel=1;dtel<15;dtel++)

```

```

{ gotoxy(1,dtel); printf("
");
}

clrscr();

printf("Technische opmerkingen:\n\n");

printf("Bereik: Kleiner dan .01 en groter dan 100 levert onbetrouwbare resultaten op.\n");

printf("    Dit door het maximum bereik van 'float-type' variabelen.\n\n");

printf("Schaal: De schaal is alleen re%oel in lineaire-ruimte afbeelding. In gekromde-\n");

printf("    ruimte afbeelding is de grootte het resultaat van een algoritme dat\n");

printf("    bepaalt in hoeverre de kromming van de 1 db's de in 3D berekende kubus\n");

printf("    van vlakke ruimte vervormd. ");

getch();


}

} while (prog>0);

}

```

Patent: Analogue computer chip

In order to optimize the software calculating the movement and character of s a new type of computer chip has been thought of. The accompanying patent for the chip is shown below.

Patent NL: 2018026: Method and system for storage and processing of data.

Inventor: Gerhard Jan Smit

Field

The invention relates to a method and a system for the processing of digital information.

Background

For the storage of data there are several systems in circulation. One can think of:

Magnetic discs: diskette, harddisk, RAID, DAS, zipdisc, jaz, minidisc;

Optical discs: compact disc, dvd, blu-ray, hd-dvd, CBHD, EVD, HVD, UMD, VMD;

Magnetic tapes: compact cassette, microcassette, dat, dcc, digital video;

Electronic: RAM, registermemory, ROM, cartridge, PROM, EPROM, EEPROM, flashmemory, memory card, solid state drive, USB-stick, Cavendish-cel, DOM, MRAM, PCRAM, VRAM;

Punched paper: punch card, punch tape;

Others: cache, NAS, SAN.

For the development of computer technology and the processing of great amounts of data there is need for powerful software and big storage volumes. Besides enlargement of storage volumes and speedup of processing (computing capacity) is reliability of the data of

great importance. The current way of storage and processing of information has its limits. Seen from this perspective a new way of data storage and processing of information is of great importance.

Resumé of the invention

The invention has a method for the processing of data, comprising:

- The retrieval of a digital data file;
- The conversion of the digital file to a real number and the generation of an analog signal that represents that real number;
- The storage of the analog signal in an analog storage unit.

In further elaboration of the invention, processing can also be effected, included:

- The conversion of the analog signal of the in an analog storage unit stored analog signal that represents the real number in a digital data file.

The invention also provides a system for processing of digital data, in which the system includes:

- An analog storage unit with an input and an output;
- A first converter configured for the conversion of a digital data file in a real number and the generation of an analog signal that represents the real number and that serves as input for the analog data storage unit;
- A second converter for the conversion of the in the analog data storage unit stored analog signal and the conversion of the real number in a digital data file.

Further elaborations of the invention are described in the conclusions to follow and will be further clarified below.

In order to generate large storage capacity and as a means for improved computing capacity, the analogue storage is proposed. For this, a binary sequence is converted to a real number. This real number is converted into an analog signal that can be stored, for example, as an electric voltage, an amperage or an electric resistance. The analog signal thus stored can be read and be converted to a binary sequence corresponding to the original binary sequence that served as the basis for the conversion to the analog signal. The information may contain text files but can also have an arithmetical meaning. The original information can contain any string of characters with any length.

With the aid of the method and the system according to the invention the following effects/benefits can be achieved:

- Large storage capacity can be generated by analog storage of encrypted information, for this purpose a binary sequence is converted to a real number;
- Large computing capacity can be generated by using analogue storage of encrypted

information, for this purpose a binary sequence is converted to a real number; Digital information can be coded to a real number and is stored on an analog storage unit as a voltage, an amperage or as a resistance; the analog signal (that represents the real number) can be read and then to the need for the benefit of the user be decoded into digital information; the selected real number is again used for this purpose and converted to a binary sequence;

The information stored in the manner described above can possibly contain text files but can also have an arithmetic meaning; the stored information relates to each random string of any length;

The invention is applicable within computers, tablets, smartphones and various other electronic facilities;

If the analogue storage units are connected in parallel and/or in series then possibilities for PU-use arise (Processor/Processing Units), ROM-use (Read Only Memory), and RAM-use (Random Access Memory), multiple circuits can lead to intelligent computational solutions.

Further elaborations are described in the subclaims. The invention will be further described below with reference to the figures.

Short description of the figures

Figure 1 shows a schematic representation of an example of a system according to the invention; Figure 2A shows a representation of redundant storage of an analog signal, the analog signal representing 0; Figure 2B shows a representation of redundant storage of an analogue signal, where the analog signal represents a value different from 0.

Fig. 1: A schematic representation of an example of a system according to the invention.

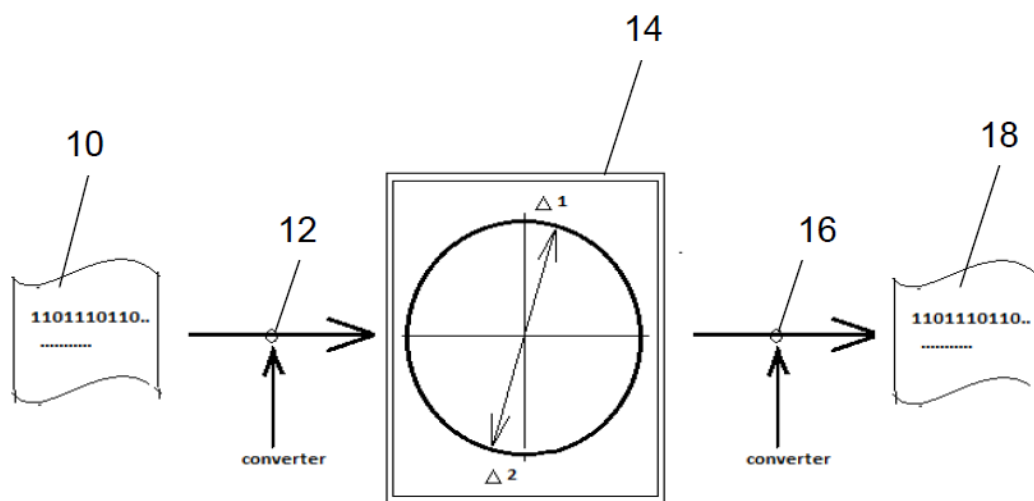


Fig. 1

Fig. 2A: A representation of redundant storage of an analog signal, the analog signal representing 0; Fig. 2B: A representation of redundant storage of an analogue signal, where the analog signal represents a value different from 0.

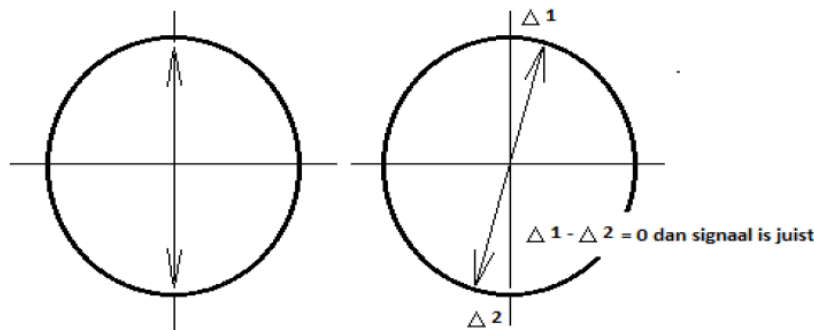


Fig. 2A

Fig. 2B

($\Delta 1 - \Delta 2 = 0$ dan signal is juist : $\Delta 1 - \Delta 2 = 0$ then signal is correct)

Detailed description

The invention relates to the analog storage of information of any string and of any length. For this, a binary sequence is converted into a real number. This real number is then converted into an analog signal that can be stored in an analogue storage unit. The analog signal may for example comprise an electric voltage, an electric current strength or an electrical resistance. The information thus stored can then be read again and converted into a binary sequence, in particular the same binary sequence that served as input and on the basis of which the analog signal was generated.

Figure 1 shows a schematic representation of a system for the processing of digital data. In it with reference numeral 10 first a digital data file is indicated in the form of a binary sequence. This binary sequence becomes converted in converter 12 into an analog signal that represents a real number. This analog signal is stored in an analog storage unit 14. In the example shown, this is a redundant analog storage unit in which the analogue signal is stored twice as $\Delta 1$ and $\Delta 2$.

The analog signal can be output from the analog storage unit 14 via an output to a second converter 16 which converts the analog signal in a second binary sequence 18 that forms a digital data file that corresponds to the first digital data file 10.

Figure 2A shows a representation of redundant storage of an analog signal, the analog signal representing 0. Fig. 2B shows a representation of redundant storage of an analogue signal, where the analog signal represents a value different from 0. The analog signal is stored twice, once as $\Delta 1$ and once as $\Delta 2$. To check if the saved analog signal is stored correctly, it can be determined whether the difference between $\Delta 1$ and $\Delta 2$ is zero (i.e., $\Delta 1 - \Delta 2 = 0$). If this is the case, it can be assumed that the conversion of the binary sequence to the analog signal is executed correctly. $\Delta 1$ and $\Delta 2$ can, for example, be determined in succession with

the same converter 12. However, it is also possible that $\Delta 1$ and $\Delta 2$ are parallel determined with two different converters 12 converting the binary file each in their own way to an analogue signal.

Binary information (binary sequence) turned into a real number can be done by means of software. The real number can have a text as origin but can also have an arithmetical meaning. The generated real number is saved via the software as a value between 0 and 1. We expressly mention that in essence any readable number can be produced and handled. The choice here to choose " $0 \leq \text{number} < 1$ " is of practical origin. The number can assume every length. This depends on the amount of original information and character of the analogue storage unit used (examples are a capacitor or a battery). The number can be expressed binary (example: 0.100001011110010101 ... etc ...), decimal (example 0.1453345673385245 ... etc ...), hexadecimal or as another numerical expression.

The software that is used to generate the real number must use a strict, set routine. This is important because the real number will eventually be translated back to the original binary information. In our example, the real number always starts with "0.". This achieves that the value of the real number is between 0 and 1.

The real number determined by the software becomes one-to-one written to an analogue medium for example as a voltage, an amperage or as a resistance. Redundancy for recording the signal makes sense. A representation of redundant storage is shown in Figure 2.

When the analogously written real number must be addressed for the purpose of reading information and this information must be made accessible for a user then a conversion must be carried out for this purpose, for example by means of software. This software applies the same strict, set routine but then in reverse order (decoding). In the manner described here large amounts of information can be recorded. The information can be used for storage or for arithmetic applications.

Information provided within computers or other electronic equipment written binary can be directly (whether or not preceded by "0." or another number) written one-to-one to an analogue medium as a voltage, an amperage or as a resistance. In this case, the software described above can be omitted. Here again, applying redundancy to the recording of the signal makes sense.

When the in this method and this system used analogue storage units are connected in parallel and/or in series then the possibilities for PU use (Processor/Processing Units), ROM use (Read Only Memory), and RAM usage (Random Access Memory) arise.

Multiple circuits can lead to intelligent computational solutions.

To produce a chip on which the analog signals can be written one can use traditional analog bipolar transistors or field-effect transistors (FET). It is useful to use FETs on a molecular scale. To this end, please refer to the article by Yoshihiro Kubozono, Keita Hyodo, Shino Hamao, Yuma Shimo, Hiroki Mori & Yasushi Nishihara (www.nature.com/scientificreports dated 6 December 2016). In the article with the title: "Transistor Properties of 2,7-DialkylSubstituted Phenanthro [2,1-b: 7,8-b'] dithiophene" is indicated that Phanancens are

very suitable molecules for use in FETs. This allows chips in which this invention is processed to be built on a small scale.

In further elaboration of the invention the following may apply:

For the purpose of analog storage, the original binary information (binary sequence) can by means of software be converted into a real number;

The generated real number may have a text as origin but can also have an arithmetic meaning consisting out of each random string of any length;

A binary sequence is written via the software as a real number;

The software that is used to generate the real number must use a strict, set routine;

The real number determined by the software is written to an analog medium or an analog storage unit in the form of an analogue signal, such as for example a voltage, a resistance or an amperage;

The writing of the real number must be completely analogous;

The recorded real number can be addressed and be made accessible for the user, here conversion of the real number to a binary sequence should take place (decoding);

Information (a sequence) that is written binary within computers or other electronic equipment can also directly be written as a real number on an analogue medium, for example as a voltage, an amperage or as a resistance. In that case, the described software can stay away;

An example of an analog storage unit is a capacitor or a battery.

The various examples described above can be combined in different manners independently of each other and with each other. The reference numbers in the detailed description and the conclusions do not limit the description of the embodiments and serve only for clarification.

Conclusions

1. A method for processing data, comprising:
 - providing a digital data file;
 - converting the digital data file to a real number and generating an analog signal that represents the real number;
 - storing the analog signal in an analog storage unit.
2. The method according to claim 1, wherein converting the digital data file to a real number and the generation of an analogue signal representing the real number includes:
 - to normalize the digital data file to a real number between 0 and 1.
3. The method according to claim 2, wherein the normalizing is being executed by 0. to place for the digital file consisting of zeros and ones.
4. The method according to any one of claims 1-3, wherein it also includes:

converting the data stored in the analog storage unit as an analog signal representing the real number in a digital data file.

5. The method according to claim 4, wherein the conversion of the, in the analog storage unit stored analog signal, that represents the real number in a digital data file, is executed by the real number that is represented by multiplying the analog signal by 10, and from the real number thus obtained, that only contains zeroes and ones, is turned into a digital file that has the same series of zeros and ones included as obtained from the real number.
6. The method according to any one of the preceding claims, wherein the analog signal is an electrical voltage, an electric current or a electrical resistance.
7. A system for processing digital data, where the system includes:
 - an analog storage unit with an input and an output;
 - a first converter configured for converting a digital file into a real number and generate an analogue signal representing the real number and serving as input for the input from the analog storage unit;
 - a second converter for converting the, into the analog storage unit stored analog signal, in a real number, and the converting of the real number into a digital data file.
8. The system according to claim 7, wherein the analog storage unit is selected from the group comprising at least one of the following components:
 - analog bipolar transistors;
 - field effect transistors (FET);
 - a capacitor; and
 - a battery.
9. The system according to claim 7 or 8, wherein the analogue storage unit is configured to store the analog signal in the shape of an electrical voltage, an electric current or a electrical resistance.

End of document Patent NL: 2018026