An interactive real-time physics software for structural analysis of space trusses

Reydleon Paiva de Paulo

Instituto Superior Técnico, University of Lisbon, Portugal (Sept 2020). reydleon@outlook.com

Abstract: Space truss is an important theme while learning the first concepts of statics. On the other hand, it has real engineering applications. An example of the application of this technique is the Double Layer Grid (DLG), which is generally the adopted solution in the roofing of factories and airport terminal halls, as it can overcome large spans. This work presents an interactive software entirely written in Python that aims to help students to understand the behavior of space truss systems. The software allows the user to design structures, analyze them, and then export the results. The CAD software was successfully created from scratch and can perform nonlinear analysis. The 3DParticleSystem software numerical method is based on the concept of physics engine. Physics engines are widely used as middleware in game engines. In commercial software for structural analysis this approach is rarely used despite exhibiting some quite convenient features: it allows for new types of analysis (namely, nonlinear and incremental analysis); also allowing to represent the time evolution of a structure once calculations are made in real-time, responding to user input. Such features can be important while learning structural engineering concepts. The work carried out here further improves the application of physics engines in the field of structural analysis: it first summarizes the implementation of the physically-based modeling/particle system dynamics; it then gives an overview of the PyQt and the Panda3D game engine, tools that were used to create an advanced GUI (Graphical User Interface) to render space trusses.

Keywords: Interactive Structural Analysis, Python 3D physics engine, Panda3D, Nonlinear Analysis, Newton's Second Law, Particle System Dynamics.

1 Introduction

Commercial structural analysis software are, usually, based on implementations of the Finite Element Method (FEM). There are, nevertheless, some numerical alternatives to FEM. One such alternative resorts to particle-based methods. These methods are widely used in games and animations development but also have applications in structural engineering problems such as structural form-finding problems [1], and others based on the dynamic relaxation method [2].

The objective of this project is to develop an interactive software that supports learning the fundamentals of structural engineering and space trusses. The developed software should have some features commonly found in advanced structural analysis software, but at the same time, be user-friendly and didactic. Thus, one possible approach for the development of this tool is to use the concepts of game engines and physics engines. One of the advantages of this approach is that it handles kinematically indeterminate structures as well as determinate ones. Moreover, it can provide real-time simulation feedback about structural performance, including internal forces and reactions to the users [3].

The PSA3D physics engine, the code developed during the course of this work, was written from scratch in Python [4] and was combined with a graphics rendering middleware created using the Panda3D game engine [5]. The result of this combination is the 3DParticleSystem software presented in this work (Figure 1). This software gives the user a unique game-like interaction where structural models respond to input in real-time.

The 3DParticleSystem can be used to instantly see the effect of removing bars and changing nodal constraints during the simulation; allowing even to analyze the instant when the structure becomes a mechanism.

The tasks carried out during this work are summarized in chronological order:

a) Review of particle system dynamics;
b) Development of a physics engine and its validation with 2D truss problems;
c) Development of a Graphical User Interface and a 3D graphical environment;
d) Extension of the numerical approach to 3D problems;
e) Comparison of the computed results with results obtained with ADINA [6].

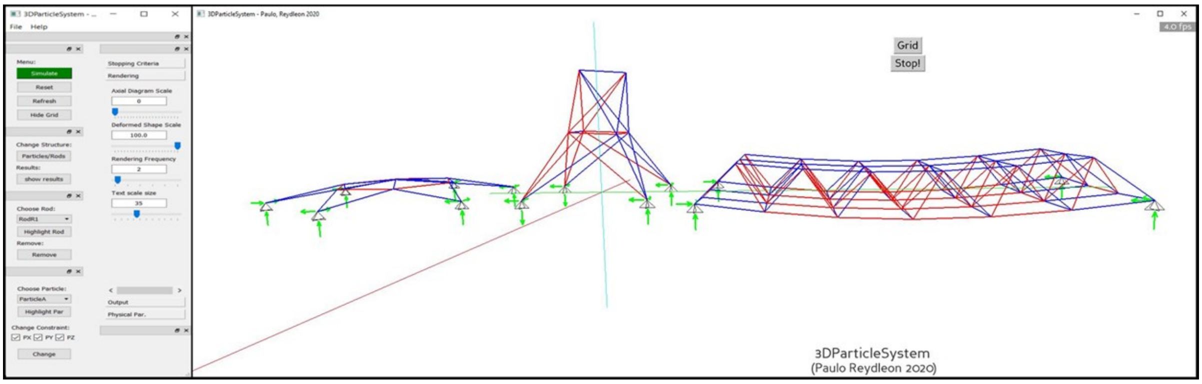

Figure 1 – The graphical output of the developed software. 3DParticleSystem software.

2 Particle System Approach 3D

PSA3D, Particle System Approach 3D, is a 3D physics engine library written from scratch in Python for structural engineering applications. It is based on particle system dynamics, which is a physically based particle model that follows the laws of classical mechanics. It is designed to solve trusses, from simple 2D trusses to complex 3D truss structures composed by materials with linear and nonlinear behaviors and different stress and strain behaviors, such as elastic and elastic-plastic behaviors. PSA3D works without having a graphical component, being the results written in the python console.

2.1 Mechanical behavior of materials

In teaching the fundamental concepts of mechanical behavior of materials in civil engineering, the materials that come to mind are concrete and steel. Concrete resists far better to compressive stresses than to tensile stresses, whereas steel is indifferent to the type of stress it is subjected to. Compressive and tensile strength tests are generally used to describe the constitutive equations of engineering materials.

Generally, plastic behavior is conveniently simplified using idealized models. The most common models are the elastic-perfectly plastic model and the bilinear hardening model. Figure 2 presents the symbolic one-dimensional models for a better understanding of this idealized model. On the left, it presents the elastic-perfectly plastic solid with the yield limit Fy; on the right, the elastic-plastic body with linear strain hardening.

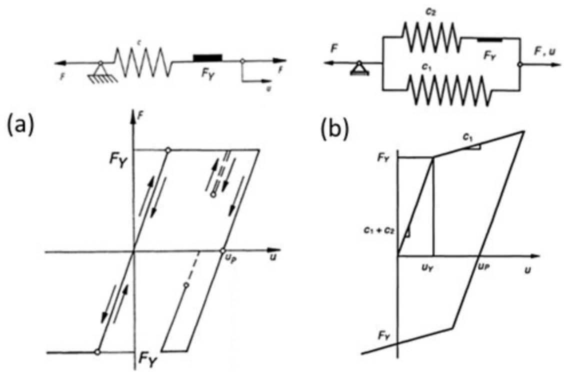

Figure 2 – Symbolic one-dimensional models: (a) The elastic-perfectly plastic solid with the yield limit Fy. (b) The elastic-plastic body with linear strain hardening. Adapted from [22].

2.2 Particle System Dynamics

The fundamentals of particle system dynamics are described by Witkin and Baraff, both from Pixar Animation Studios, in [7][8], and are commonly used in physics-based animation. The fundamental principle of this approach is to discretize the structural system into particles and rods, as shown in Figure 3.

Particles are linked to other particles by means of massless springs (in the case being treated here, mostly elastic truss members) that are in static equilibrium during motion. The forces acting on a particle may be external forces (loads) or internal forces. Equilibrium is enforced on each particle, resulting in the continuous balance of internal nodal force and external force. In this approach, the differential equations of motion of a collection of particles are solved using a time-step simulation.

2.2.1 Newton's second law

The basic issue in dynamics problems is solving the fundamental equation of dynamics, also known as Newton's second law. First, to establish a consistent reading with recent literature, the notation used here is that presented in [9] by Ying Yu. Thus, the motion of an arbitrary particle i follows Newton's second law and can be expressed as Equation (1).

$$\mathrm{m_i}\begin{bmatrix}\ddot{\mathrm{x}}_{i,\mathrm{x}}\\ \ddot{\mathrm{x}}_{\mathrm{i,y}}\\ \ddot{\mathrm{x}}_{i,\mathrm{z}}\end{bmatrix} = \begin{bmatrix}F_{i,x}^{ext}\\ F_{i,y}^{ext}\\ F_{i,z}^{ext}\end{bmatrix} + \begin{bmatrix}F_{i,x}^{int}\\ F_{i,y}^{int}\\ F_{i,z}^{int}\end{bmatrix} + \begin{bmatrix}F_{\mathrm{i,x}}^{\mathrm{reaction}}\\ F_{\mathrm{i,y}}^{\mathrm{reaction}}\\ F_{\mathrm{i,z}}^{\mathrm{reaction}}\end{bmatrix} = \begin{bmatrix}F_{i,x}^{unb}\\ F_{i,y}^{unb}\\ F_{i,z}^{unb}\end{bmatrix} \quad (1)$$

where $\mathrm{m_i}$ is the mass of particle i; x_{i} is the displacement vector; $\ddot{x}_i$ is the acceleration vector. $\mathrm{F}_i^{\mathrm{ext}}$, $\mathrm{F}_i^{\mathrm{int}}$, $\mathrm{F}_i^{\mathrm{reaction}}$, and $\mathrm{F}_i^{\mathrm{unb}}$ are the external force, internal nodal force, support reaction contribution, and unbalanced force, respectively.

2.2.2 Particle definition and particle mass

The mass of the structure, that is the mass of the rods/bars composing the structure, is divided proportionally between the particles, and these can simulate the characteristics of the structure. Particles have mass and are connected by massless rod elements. The mass of each rod element $\mathrm{m_{rod}}$ is equally divided between the particles that define the rod and this contribution is defined as an equivalent mass m_α. Thus, the mass of a given particle i is defined as:

$$\mathrm{m_i} = \sum_{\alpha=1}^{\mathrm{n}} \mathrm{m}_\alpha \quad (2)$$

$$\mathrm{m}_\alpha = 0.5 \times \mathrm{m_{rod}}\ ;\ \mathrm{m_{rod}} = \lambda \cdot \mathrm{L} \quad (3)$$

where n is the number of rods connected to the particle i, m_α is the mass contributed from each rod element and assigned to particle i. Moreover, λ is the mass per unit length of the rod element, and L is the length of the rod element.

2.2.3 Calculation of the internal nodal force

In this work, two approaches are presented to calculate the internal nodal force: one considering small displacements and another considering large displacements.

2.2.3.1 Small Displacements

The internal force in a given 3D rod k, connecting particles 1 and 2, can be written as [7]:

$$f_{\mathrm{k,local\ axis}}^{\mathrm{int}} = \varepsilon_{\mathrm{n+1}} \cdot \mathrm{EA};\ \varepsilon_{\mathrm{n+1}} = \frac{\mathrm{L_{n+1}} - \mathrm{r}}{\mathrm{r}} \quad (4)$$

$$L_{n+1} = \|\mathrm{X}_{2'} - \mathrm{X}_{1'}\|\ ;\ \mathrm{r} = \|\mathrm{X}_2 - \mathrm{X}_1\| \quad (5)$$

where r is the rest length of a rod k. $\mathrm{L_{n+1}}$ and $\varepsilon_{\mathrm{n+1}}$ are the length and strain at the given time, respectively. E is the Young's modulus, and A is the cross-sectional area of rod k.

2.2.3.2 Large Displacements

Large displacements analysis can be achieved by using the approach described in [9][10][11] in which Yu Yung presents the Finite Particle Method, suitable for modeling geometric nonlinearity behavior.

The axial force in rod k can be written as:

$$f_{\mathrm{k,local\ axis}}^{\mathrm{int}} = f_{\mathrm{n+1}} = f_{\mathrm{n}} + \Delta f_{\mathrm{n}} = \left(\sigma_{\mathrm{n}}\mathrm{A_n} + \frac{\mathrm{E_n A_n}}{\mathrm{L_n}}\Delta\mathrm{L}\right)\mathrm{e}_{1'2'} \quad (6)$$

where f_{n} is the axial force of rod k at time t_{a}; Δf_{n} is the incremental axial force of rod k at time $\mathrm{t_b}$; σ_{n} is the axial stress at time $\mathrm{t_a}$; $\mathrm{A_n}$ is the cross-sectional area of rod k; E_n is the Young's modulus; $\Delta\mathrm{L}$ is length variations of rod k between time $\mathrm{t_a}$ and $\mathrm{t_b}$; $\mathrm{L_n}$ is the length of rod k at time $\mathrm{t_a}$; $\mathrm{e}_{1'2'}$ is the directional vector of rod k at time $\mathrm{t_b}$.

2.2.3.3 Internal nodal force

The rod's internal forces can be transformed from principal axes to the global axis with the following linear transformation:

$$f_{\mathrm{k}}^{\mathrm{int}} = T^T f_{\mathrm{k,local\ axis}}^{\mathrm{int}} = \left[f_{\mathrm{k,x}}^{\mathrm{int}}\ f_{\mathrm{k,y}}^{\mathrm{int}}\ f_{\mathrm{k,z}}^{\mathrm{int}}\right]^{\mathrm{T}} \quad (7)$$

The relationship between the rod's internal forces and the forces acting on each particle is given by the particle 1 (linked to the beginning of the rod) receiving the force $f_{\mathrm{k}}^{\mathrm{int}}$ and particle 2 (at the end of the rod) receiving the same force in the opposite direction, according to the static equilibrium condition [10]. Then, it can be stored in a 1x6 vector as follows:

$$f_{\mathrm{k}}^{\mathrm{int}} = \left[f_{\mathrm{k,1}}^{\mathrm{int}^{\mathrm{T}}}\ \mid\ f_{\mathrm{k,2}}^{\mathrm{int}^{\mathrm{T}}}\right] = \left[f_{\mathrm{k}}^{\mathrm{int}^{\mathrm{T}}}\ \mid\ -f_{\mathrm{k}}^{\mathrm{int}^{\mathrm{T}}}\right] \quad (8)$$

where $f_{\mathrm{k,1}}^{\mathrm{int}^{T}} = f_{\mathrm{k}}^{\mathrm{int}^{\mathrm{T}}}$ and $f_{\mathrm{k,2}}^{\mathrm{int}^{T}} = -f_{\mathrm{k}}^{\mathrm{int}^{\mathrm{T}}}$ are the forces at 1 and 2, respectively.

Finally, the internal nodal force of the particle i can be identified by the summation of the internal/axial forces from all rods connected to it and can be given as follows:

$$F_i^{int} = \begin{bmatrix} F_{i,x}^{int} \\ F_{i,y}^{int} \\ F_{i,z}^{int} \end{bmatrix} = \begin{bmatrix} f_{1,x}^{int} \\ f_{1,y}^{int} \\ f_{1,z}^{int} \end{bmatrix} + \cdots + \begin{bmatrix} f_{n,x}^{int} \\ f_{n,y}^{int} \\ f_{n,z}^{int} \end{bmatrix} = \sum_{k=1}^{n} \left(\begin{bmatrix} f_{k,x}^{int} \\ f_{k,y}^{int} \\ f_{k,z}^{int} \end{bmatrix} \right) \tag{9}$$

where $\mathrm{F}_{\mathrm{i}}^{\mathrm{int}}$ is the internal nodal force acting on the particle i, f_k is the equivalent internal forces provided by rod k connected to particle i, and n is the number of rods connected to particle i.

2.2.4 Convergence and stability

The main idea of this approach is to "follow" the particle motion caused by the unbalanced forces. Convergence is usually achieved by damping the nodal movements using artificial viscous damping [7] or considering a form of kinetic energy damping [12].

2.2.4.1 Particle System Approach

In the Particle System Approach, the axial force in each rod results by adding the rod's damping forces with its internal forces.

The rod's damping force, also described as the ideal viscous drag in Witkin's notation [7], can be considered as a way of extracting energy from the system to bring it to rest. It can be expressed as:

$$f_{\mathrm{k,local\ axis}}^{\mathrm{dmp}} = \mathrm{c} \cdot \Delta v_{local\ axis} = \mathrm{c} \cdot (v_{2,local} - v_{1,local}) \tag{10}$$

where c is the damping coefficient, $v_{\mu,local}$ is the velocity on the local axis of edge μ linking nodes 1 and 2.

In this approach, the axial force of rod k can be written as:

$$f_{\mathrm{k,local\ axis}}^{axial} = f_{\mathrm{k,local\ axis}}^{\mathrm{int}} + f_{\mathrm{k,local\ axis}}^{\mathrm{dmp}} = \mathrm{EA} \cdot \varepsilon + f_{\mathrm{k,local\ axis}}^{\mathrm{dmp}} \tag{11}$$

where $f_{\mathrm{k,local\ axis}}^{\mathrm{axial}}$ is the resulting internal force. $f_{\mathrm{k,local\ axis}}^{\mathrm{int}}$ is the internal force given by Equation (4) and $f_{\mathrm{k,local\ axis}}^{\mathrm{dmp}}$ is the damping force.

2.2.4.2 Kinetic Damping Approach

This method can be divided into two stages: the first stage consists of solving the equation of motion. The second stage consists of correcting the positions of all the particles whenever a peak of kinetic energy occurs.

2.2.4.2.1 Stage 1 - Numerical integration

The first stage is composed of Equation (12) and Equation (13), which were deduced by Barnes in [12]. These make it possible to compute the velocities and positions of the particles.

The velocity at time $t + \frac{\Delta t}{2}$ is given by:

$$v_i^{n+\frac{1}{2}} = v_i^{n-\frac{1}{2}} + \Delta t \cdot \frac{F_i^{unb}}{m_i} \tag{12}$$

The new coordinate projected at time $t + \Delta t$ becomes:

$$x_i^{n+1} = x_i^n + \Delta t \cdot v_i^{n+\frac{1}{2}} \tag{13}$$

2.2.4.2.2 Stage 2 – Kinetic Damping approach

Generally, the kinetic energy of undamped oscillations of the structure is traced when a peak in the total kinetic energy of the system is detected; consequently, all velocities are set to zero.

The kinetic energy of the whole structure is calculated as follows in [13] and can be written as:

$$K^{n+1} = \frac{1}{2} \sum_{i=1}^{q} m_i \left(v_i^{n+\frac{1}{2}} \right)^2 \tag{14}$$

where K^{n+1} is the kinetic energy of the system of particles. m_i is the mass of particle i. $v_i^{n+\frac{1}{2}}$ is the velocity of particle i and q is the number of particles.

In Kinetic Damping approach formulation, a decrease of the kinetic energy indicates that a peak has been passed [12][13]. At this time, the stored coordinates are x_i^{n+1}. For simplification, it is assumed that the kinetic energy attains its peak at $t - \frac{\Delta t}{2}$. All particles coordinates must be then recalculated/restarted at the peak instant and can be evaluated with:

$$x_i^{n-\frac{1}{2}} = x_i^{n+1} - \frac{3}{2} \Delta t\, v_i^{n+\frac{1}{2}} + \frac{\Delta t^2}{2} \cdot \frac{F_i^{unb}}{m_i} \tag{15}$$

When the analysis is restarted, the velocities must be computed in the middle point of the previous interval. Thus, the velocity is:

$$v_i^{n+\frac{1}{2}} = \frac{1}{2} \cdot \frac{F_i^{unb}}{m_i} \cdot \Delta t \tag{16}$$

where all unbalanced forces, F_i^{unb}, must be evaluated at the $x_i^{n-\frac{1}{2}}$ position using Equation (15).

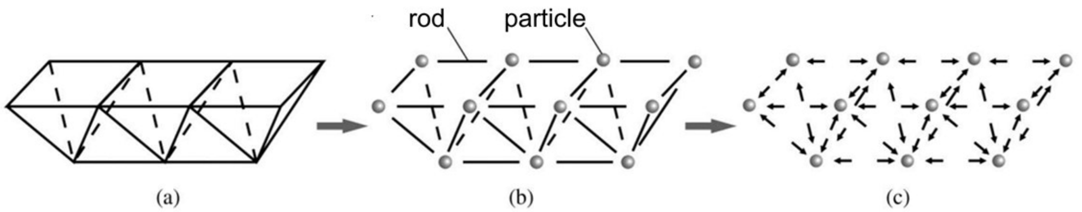

Figure 3 – (a) space truss structure; (b) discretization of the structure by particles and elements; (c) particles and forces. Adapted from [9].

2.3 Numerical implementation of the PSA3D physics engine

2.3.1 Numerical method

The Numerical method is the implementation of the Particle System approach (PS), which consists of solving Newton's second law for a set of particles by numerical integration. It consists of a loop that computes the numerical process until the stopping criterion is met.

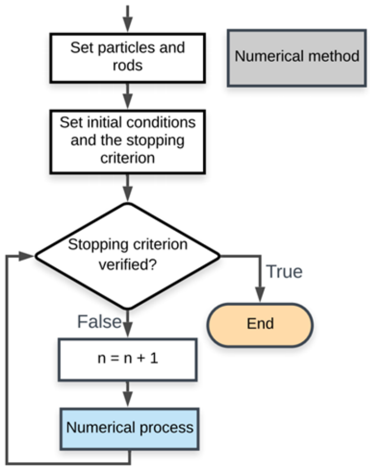

Figure 4 – Numerical method flow chart.

As follows, two approaches are presented for the implementation of the Numerical process. Approach 1 considers the small displacements formulation and a damping force approach; Approach 2 considers the large displacements formulation and a kinetic energy damping approach.

2.3.2 Numerical process – Approach 1

The Numerical process for Approach 1 consists of looping both list of particles and list of rods, as presented in the figure below. First, the numerical integration is implemented for each particle by applying the trapezoidal rule. Then, the axial force is calculated for each rod.

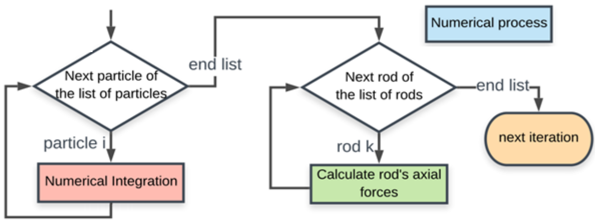

Figure 5 – Numerical process – Approach 1.

2.3.2.1 Numerical integration of the equation of motion

There are several numerical integration techniques used for solving the equation of motion [14]. The most common are the Euler's Method, the Verlet Method, and the Runge-Kutta Method [15]. Trapezoidal rule is a refinement of Euler's method; it consists of using the mean velocity during the interval to obtain a new position [15] and its application can be summarized as:

$$v_{n+1} = v_n + a_n \Delta t \tag{17}$$

$$x_{n+1} = x_n + \frac{1}{2}(v_n + v_{n+1})\Delta t \tag{18}$$

All the steps must be repeated for the other directions.

The Numerical Integration of the Equation of Motion algorithm in Approach 1 consists of applying the trapezoidal rule. In this way, the unbalanced forces at particle are transformed into acceleration, and then, into motion of the particle. In the first step, the unbalanced forces (F_i^{unb}) are calculated, and then the acceleration (a_i^n). Later, velocities (v_i^{n+1}) and positions (x_i^{n+1}) are calculated using Equation (17) and Equation (18), respectively.

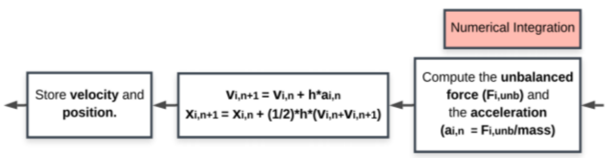

Figure 6 – Numerical Integration of the Equation of motion – Trapezoidal rule flow chart.

2.3.2.2 Rod's axial force

Approach 1 considers the formulation for small displacements. In this case, the axial force in each rod results by adding the rod's damping forces with its internal forces, Equation (4). This quantity can be computed with Equation (11) and implemented as follows in Figure 7.

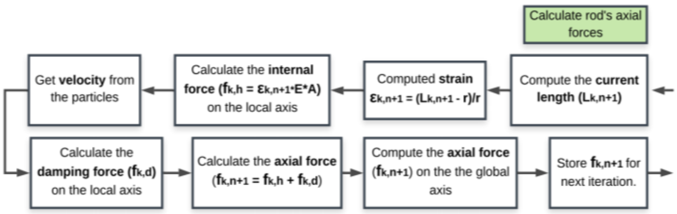

Figure 7 – Flow chart for the calculation of rods' axial forces in Approach 1.

2.3.3 Numerical process – Approach 2

Similarly to Approach 1, the Numerical process for Approach 2 also consists of looping both the list of particles and the list of rods. Besides, it has an extra step named the Kinetic Damping approach, summarized in Figure 8.

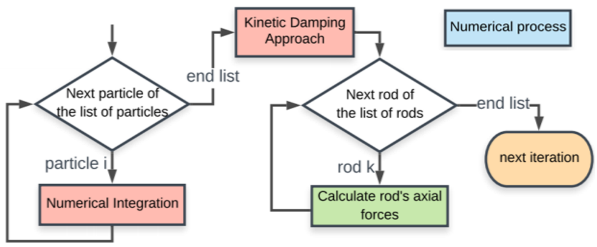

Figure 8 – Numerical process – Approach 2.

2.3.3.1 Stage 1 – Numerical integration

The stage 1 consists of applying equations (12) and (13), as presented in Figure 9. As described in [12], the first iteration may be set as a peak and the velocity can be calculated with Equation (16).

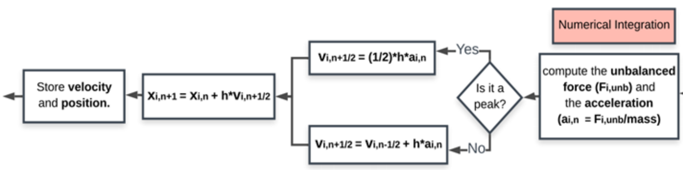

Figure 9 – Stage 1 of the Kinetic Damping approach - Numerical integration flow chart.

2.3.3.2 Stage 2 – Kinetic Damping Approach

The second stage of the Kinetic Damping approach consists of re-initializing the velocities at kinetic energy peaks by applying equations (14) and (15). Figure 10 summarizes the implementation of stage 2.

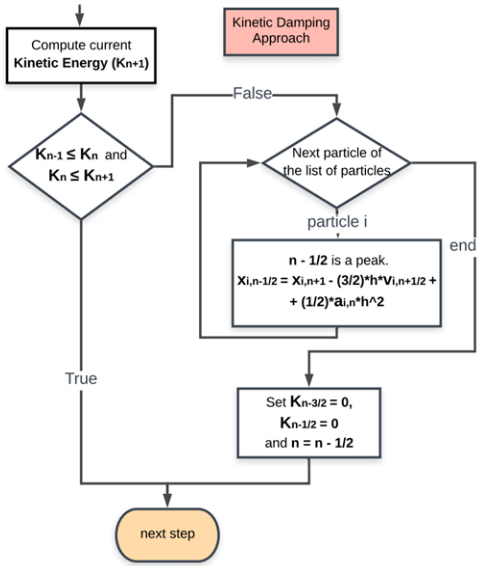

Figure 10 – Stage 2 of the Kinetic Damping approach – flow chart.

2.3.3.3 Rod's axial force

Approach 2 considers the formulation for large displacements. Thus, the rod's axial force can be given by Equation (6), which is conveniently recalled here as Equation (19). This quantity can be implemented as follows in Figure 11.

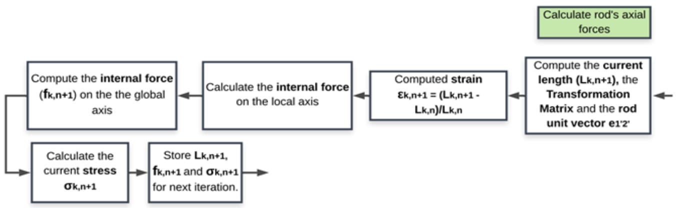

Figure 11 – Flow chart for the calculation of rods' axial forces in Approach 2.

$$f^{\mathrm{int}}_{\mathrm{k,local\ axis}} = \left(\sigma_{\mathrm{k,n}} A_{\mathrm{n}} + E_{\mathrm{n}} A_{\mathrm{n}} \varepsilon_{\mathrm{k,n+1}}\right) e_{1'2'}$$
$$\varepsilon_{\mathrm{k,n+1}} = \frac{\Delta L}{L_{\mathrm{k,n}}} = \frac{L_{\mathrm{k,n+1}} - L_{\mathrm{k,n}}}{L_{\mathrm{k,n}}} \quad (19)$$

2.4 Stopping criterion

The stopping criterion used in this work is:

$$\left|F_i^{unb}\right| \leq \ Tolerance \quad (20)$$

$$F_i^{\text{unb}} = \ \mathrm{F}_i^{\text{ext}} + \ \mathrm{F}_i^{\text{int}} + \ \mathrm{F}_i^{\text{reaction}} \quad (21)$$

where $\mathrm{F}_i^{\text{unb}}$ is the unbalanced force in axis i, $\mathrm{F}_i^{\text{ext}}$ is the applied external force, $\mathrm{F}_i^{\text{int}}$ is the internal nodal force and $\mathrm{F}_i^{\text{reaction}}$ is the contribution of the support reaction.

3 Implementation

The engine developed by Disney Interactive to create one of its attractions later became the Panda3D game engine. Before that, some popular commercial games such as Toontown (2003), Pirates of the Caribbean Online (2007), and A Vampyre Story (2008) were developed using this tool [16].

Although Panda3D has many features, the use of this framework in the 3DParticleSystem software is minimal as is not intended to limit the development of this software to the game engine used in this release. The developed CAD software is based on rendering a set of organized lines, which makes it possible to render arrows, circles, grids, nodes, and bars — then building all the needed modules to represent structures with complex geometries. The Panda3D framework is then combined with the PyQt framework [17], which is responsible for the Graphical User Interface, managing the main features available to the user, such as buttons and menus.

4 Modeling

This modeling chapter is divided into three parts. The first and second parts consist of applying Approaches 1 and 2 of the PSA3D physics engine to solve two problems of space trusses with different support and loading conditions. The last part presents a useful feature of the 3DParticleSystem software, namely the possibility of removing rods during the simulation.

4.1 Numerical Simulation and Software output

This section starts making an analysis of small displacements using Approach 1. In this case, a 9-bar space truss is solved. For this, it is required to set the time-step $\Delta t \ = \ 2 \times 10^{-4}\ s$ and the damping coefficient $c \ = 200 \ kN \cdot s/m$.

Later, this section presents the application of the developed software to solve a structure considering large displacements. In this case, Approach 2 is used to resolve a double layer grid. The same time-step is set for this analysis. For both cases, the stopping criterion is set as $\left|F_i^{unb}\right| \leq 10^{-6}$ for each axis.

4.1.1 Approach 1 -9-bar Space Truss

In order to verify the 3DParticleSystem software in three dimensions, a space truss system with 9 bars has been analyzed. The truss has a ball-and-socket at nodes A, B, C, and D, as shown in Figure 12. All the rods are composed with the same cross-sectional area with $\mathrm{A} = 20\,\mathrm{cm}^2$; the mass per unit length of the rod element is $\lambda = \rho A = 15.8 \ kg/m$ and modulus of elasticity of $E = 200\ GPa$. Furthermore, Figure 12 also presents the loading case.

The computed axial forces are summarized in the table below. The 3DParticle System takes 1187 iterations to meet the stopping criterion. The maximum relative difference, Equation (22), is around $4.86 \times 10^{-3}\,\%$ when compared with ADINA results (small displacements) as the reference value (u_{ref}). This level of accuracy in the application of the 3DParticle System to the analysis of 3D trusses is quite good and serves as a validation of the developed code.

$$RDif = \frac{|\Delta u|}{\left|u_{ref}\right|} = \frac{\left|u_i - u_{ref}\right|}{\left|u_{ref}\right|} \quad (22)$$

Table 1 – 9-bar space truss. Axial forces.

Rods	Axial Force (kN)		
	ADINA	**3DParticle System**	**RDif (%)**
1	61.49	61.49	1.40E-3
2	-90.00	-90.00	5.57E-4
3	11.18	11.18	1.36E-3
4	0.00	0.00	-
5	-11.46	-11.46	4.86E-3
6	-33.54	-33.54	1.75E-4
7	-14.14	-14.14	9.63E-4
8	-40.00	-40.00	6.26E-7
9	-5.00	-5.00	1.92E-3

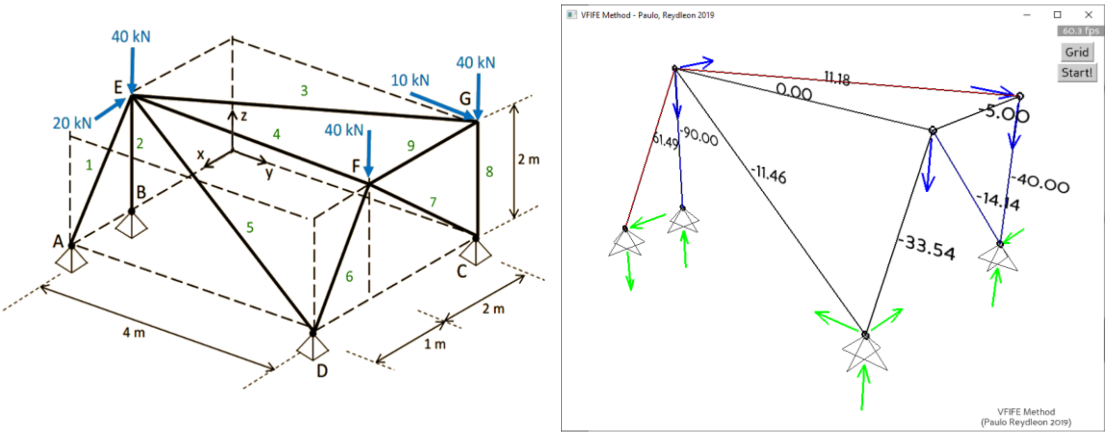

Figure 12 – 9-bar space truss (left), adapted from [23]. *Axial force results (right).*

4.1.2 Approach 2 - Double Layer Grid

The Double Layer Grid (DLG), or flat surface space frame, is one of the most common examples of space structures. DLG is a horizontal slab usually composed of interconnected square pyramids and tetrahedral tubular steel struts [18].

The DLG used in this case study is based on the experimental model analyzed by Vendrame in [19]. The general scheme of the prototypes tested is shown in Figure 14. This structure is usually denoted as the square-on-square offset space grid system. In this case, it is made up of $2.5m \times 2.5m$ orthogonal square pyramids with a height of $1.5m$. The structures are supported on the four vertices with a span of $7.5m$.

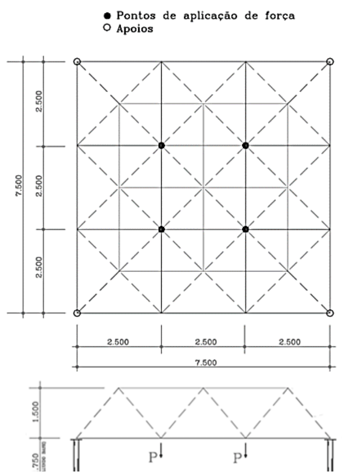

Figure 13 – Double Layer Grid (DLG) analyzed by Vendrame.

A simplification is here considered as only one of the four supports is a ball and socket support, and all the other three supports are roller supports. As the load case for this analysis, a total of $16000\ kN$ is applied over four nodes highlighted in Figure 14.

As with the last model, all rods are composed of the same cross-sectional area with $\mathrm{A} = 20\ \mathrm{cm}^2$, the mass per unit length of the rod element is $\lambda = \rho A = 15.8\ kg/m$, and elastic modulus of $\mathrm{E} = 200\ \mathrm{GPa}$.

4.1.2.1 Stress results

Stress results are summarized and compared with ADINA analysis in the tables below. Thus, the maximum compression stress occurs in the diagonal Rod 72 with approximately $6001.13\ MPa$. The maximum tensile stress occurs in the lower grille flange 2 with approximately $4852.98\ MPa$. Besides, the maximum relative difference is around 0.175% (Rod 27). The figures below show the graphical result of the DLG in different views. A simple color code indicates that red bars are subject to tensile stresses, and the blue bars are subject to compressive stresses.

Table 2 – Double Layer Grid (DLG). Axial forces.

Rods	Axial Force (kN)		
	ADINA	3DParticle System	RDif (%)
1	3428.79	3428.50	8.40E-3
2	4852.98	4850.46	5.19E-2
8	2039.07	2041.80	1.34E-1
16	267.60	267.39	8.00E-2
27	-3050.09	-3044.75	1.75E-1
36	-5187.12	-5190.09	5.72E-2
38	1124.88	1123.10	1.58E-1
39	-1557.57	-1554.95	1.68E-1
44	3377.71	3383.13	1.60E-1
72	-6001.13	-6001.69	9.26E-3

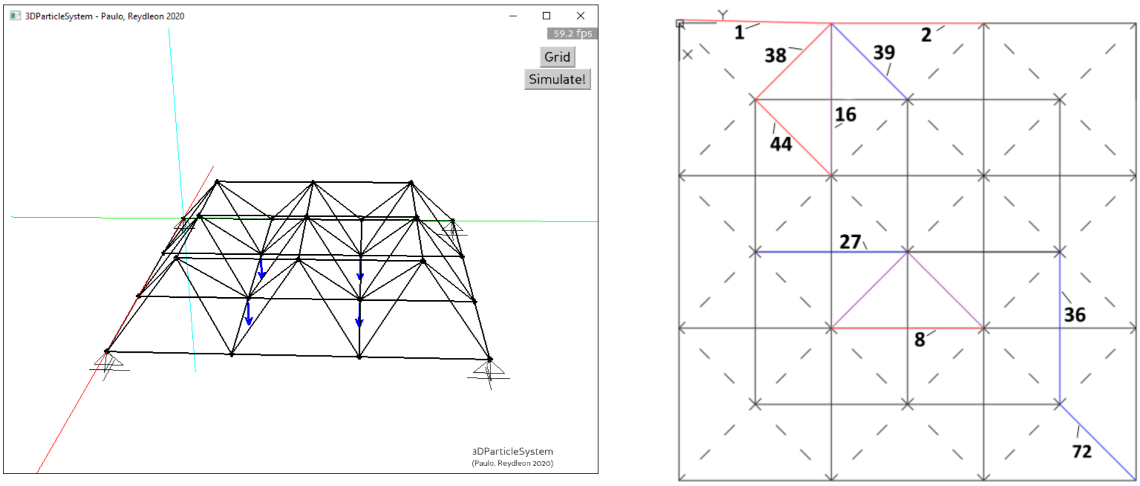

Figure 14 – Double Layer Grid (DLG). Axisymmetric view of the DLG in the developed software (left). Top view of the DLG showing analyzed rods (right).

4.2 Changing the model during the simulation

The work carried out by Martini with Arcade software [20][21] has proved to be particularly useful in teaching unstable structures and large displacements. In this way, real-time interactive physics software for structural analysis can help students understand some fundamental concepts of structural engineering, such as statically determinate (isostatic), statically indeterminate, and unstable structures (mechanisms). These topics can be easily understood using some features of the 3D Particle System software.

4.2.1 Analyzing a Double Layer Grid and removing rods

The complexity of structural analysis increases when analyzing a Double Layer Grid (DLG). Thus, this is commonly the subject of space truss analysis; however, the interpretation of these results may not be trivial due to their spatial geometry. Thus, the user can benefit from the use of 3DParticleSystem software bar-removal feature.

First, Rod 16 is removed, followed by rods 57 and 58, all of which are rods with zero axial force, as shown in the figure below. Removing these three bars does not cause large motions in the particles, just enough to redirect the unbalanced loads at each node. Thus, the stress and strain results remain as in the initial stage of this structural system. Then another nine rods corresponding to zero-force members are also removed. An impressive graphical result is that this problem becomes like a typical form-finding problem. In both cases, the structure system continually tries to find its equilibrium form by changing the position of the particle system and equilibrating the applied vertical forces.

Later, by removing a rod with considerable axial force such as Rod 72, the collapse is instantaneous. The figure below shows the resulting computed mechanism.

From the analysis presented above, it is possible to highlight some features of the 3DParticleSystem, along with the advantages of a particle-based method in structural engineering applications. The 3DParticleSystem software is proving to be an interesting teaching tool with the potential to analyze various structures.

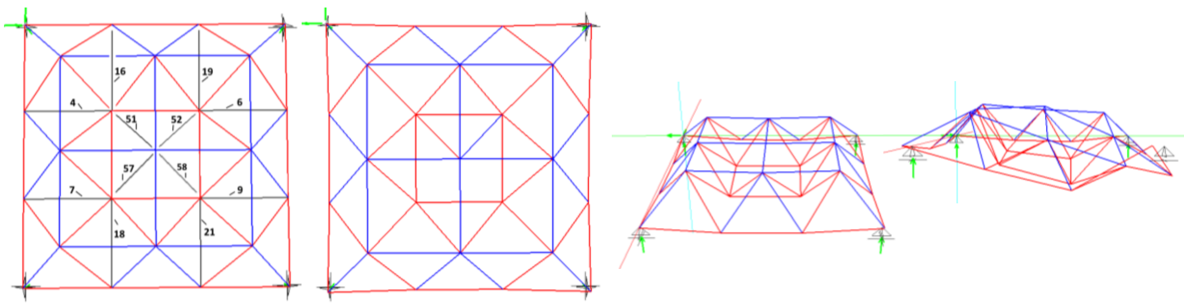

Figure 15 – Left to right. The labels of the removed rods. DLG results after rods removal, a top view, and axisymmetric view, and a mechanism.

5 Conclusions

In this document, the Particle System approach and its implementation together with a 3D graphical environment have been described.

The 3D graphical environment was developed using Python and two cross-platform frameworks, PyQt and Panda3D. The first is related to the user interface (UI); the second is related to the 3D computer graphics and the canvas.

The analyses presented in this document show good results, especially for the Double Grid Layer (DLG) case. Moreover, this software contributes to the visual perception of these structures. In this way, the developed software may help students understand some fundamental concepts of structural engineering.

As future work, there are aspects of the graphics rendering engine, and of the developed physics engine, in particular, the numerical methods used, that may be improved upon. Briefly, the possibility of analyzing frames as well as trusses frame analysis and of using other physically nonlinear materials would be useful. Also, improvements should be made to the actual code and the usability of the system.

References

[1] A. Kilian and J. Ochsendorf, “Particle-spring systems for structural form finding,” *J. Int. Assoc. Shell Spat. Struct. - IASS*, vol. 46, no. 148, pp. 77–84, 2005.

[2] A. Day, *An Introduction to Dynamic Relaxation*. 1965.

[3] C. T. Mueller, “Computational Exploration of the Structural Design Space. PhD. thesis MIT,” 2014.

[4] G. van Rossum, “Python,” 2018. [Online]. Available: https://python.org.

[5] “Panda3D.” [Online]. Available: https://www.panda3d.org/.

[6] “ADINA.” [Online]. Available: http://www.adina.com/. [Accessed: 30-Oct-2019].

[7] A. Witkin, “Physically Based Modeling: Principles and Practice,” 1997. [Online]. Available: http://www.cs.cmu.edu/~baraff/sigcourse/. [Accessed: 01-Sep-2019].

[8] D. Baraff, “Rigid Body Dynamics - SIGGRAPH 2001 COURSE NOTES,” 2001. [Online]. Available: http://www.cs.cmu.edu/afs/cs/user/baraff/www/sigcourse/. [Accessed: 01-Sep-2019].

[9] Y. Yu, G. H. Paulino, and Y. Luo, “Finite Particle Method for Progressive Failure Simulation of Truss Structures,” *J. Struct. Eng.*, vol. 137, no. 10, pp. 1168–1181, 2011.

[10] Y. Yu, P. Xia, and C. Yang, “Form finding and collapse analysis of cable nets under dynamic loads based on finite particle method,” *C. - Comput. Model. Eng. Sci.*, vol. 117, no. 1, pp. 73–89, 2018.

[11] Y. Yu and Y. Luo, “Finite particle method for kinematically indeterminate bar assemblies,” *J. Zhejiang Univ. A*, vol. 10, no. 5, pp. 669–676, 2009.

[12] M. R. Barnes, “Form Finding and Analysis of Tension Structures by Dynamic Relaxation,” *Int. J. Sp. Struct.*, vol. 14, no. 2, pp. 89–104, 1999.

[13] M. Rezaiee-Pajand and H. Rezaee, “Fictitious time step for the kinetic dynamic relaxation method,” *Mech. Adv. Mater. Struct.*, vol. 21, no. 8, pp. 631–644, 2014.

[14] H. P. Gavin, “Numerical Integration in Structural Dynamics,” *CEE 541. Struct. Dyn.*, 2014.

[15] H. Gould, J. Tobochnik, and W. Christian, *Introduction to Computer Simulation Methods: Applications to Physical Systems third edition*. 2005.

[16] GameDesigning, “In-Depth Panda3D Engine Review.” [Online]. Available: https://www.gamedesigning.org/engines/panda3d/. [Accessed: 29-Aug-2019].

[17] “Pyqt.” [Online]. Available: https://riverbankcomputing.com. [Accessed: 29-Aug-2019].

[18] T. T. Lan, “Structural Engineering Handbook - Space Frame Structures,” 2005.

[19] A. M. Vendrame, “Contribution to the study of lattice domes using steel tubular elements (In Portuguese) - MSc Thesis,” *EESC - USP*, p. 139, 1999.

[20] K. Martini, “Computational Issues in Non-linear Structural Analysis Using a Physics Engine,” in *Proceedings of the ASCE 17th Analysis and Computation Conference*, 2006.

[21] K. Martini, “Arcade - Interactive Non-linear Structural Analysis and Animation.” [Online]. Available: http://web.arch.virginia.edu/arcade/. [Accessed: 22-Oct-2019].

[22] F. Ziegler, *Mechanics of Solids and Fluids*, vol. 15, no. 2. 1995.

[23] Universiti Teknologi Malaysia, “Space Truss - Lecture 8,” *Chapter 6 - Space Trusses*. [Online]. Available: http://engineering.utm.my/civil/chapter-6/. [Accessed: 29-Aug-2019].