International Conference on Control, Engineering & Information Technology (CEIT'14)
Proceedings – Copyright IPCO-2014
ISSN 2356-5608

Static analysis of eolicblade through finite element method and OOP C++

MateusDantas, Lucas Felix[1], Raimundo M. junior[2]

[1]UFPB-CEAR-LMPD, Laboratory of Numerical Methods and Distribuited Processing

[2]UFPB-CEAR-DEER, Department of Renewable Energy Engineering

mateus.dantas@cear.ufpb.br;lucas.felix@cear.ufpb.br;jrmenezes@cear.ufpb.br

Abstract. This works deals with a description of aelastic analysis of a Eolic blade(preprocessing, processing and post-processing stages). The eolicbladegeometry, is approximate by flat finite elements in which the membrane effects, are evaluated using the FF (Free Formulation) finite element and the flexure effects, are calculated using DKT (Discrete Shear Triangle) finite element. The pre-processing stage is implemented using OpenGL library, to provide the graphical construction for geometry, mesh orientation, and other requirements of the finite element model. For the processing stage, is built a specific dll library implemented in C++ language for the FF and DKT elements analysis. The post-processing stage, has built using specific dialogs to present all resultsin the graphic interface, where are shown the static displacements of the eolicblade model.

Keywords: Shell, FEM, Dkt, Ff.

1 Introduction

One of the most important problems involving generation of energy through wind source is the structural study of the blades.These structures have a high degree of complexity and its construction in general is based on international norms (Expensive experimental tests) for certification of their safety in use. One of the potential tools to study and design of eolic blades are the numerical methods (FDM, FEM, BEM, etc). In this paper it is described an attempt to use the OOP C++ and finite element method numerical capabilities to build a tool related to analysis of eolic blades with geometry approximated by triangular flat elements. In addition, standard drawing functions from OpenGL libraryare used to provide a more friendly and efficient pre-processingtoinput element geometry, mechanical properties, element connectivity, and boundary constraints. In the processing stage the stiffness matrices related to membrane and bending effectand equivalent nodal forces (wind pressure) are evaluated by functions written in oriented object language C++ and compiled in one dll called only in the process of analysis. Finally, in the post-processing stage, all results obtained in the analysis process are shown in specific dialogs box.

2 Pre-processing

The pre-processing is the stage where input data (such as geometry, node and element numbering, mechanical properties, boundary conditions, loading) are set to perform the calculation of the discretized problem. A full discretized model of a wind generator in the tool of analysis is shown in Fig. 1.

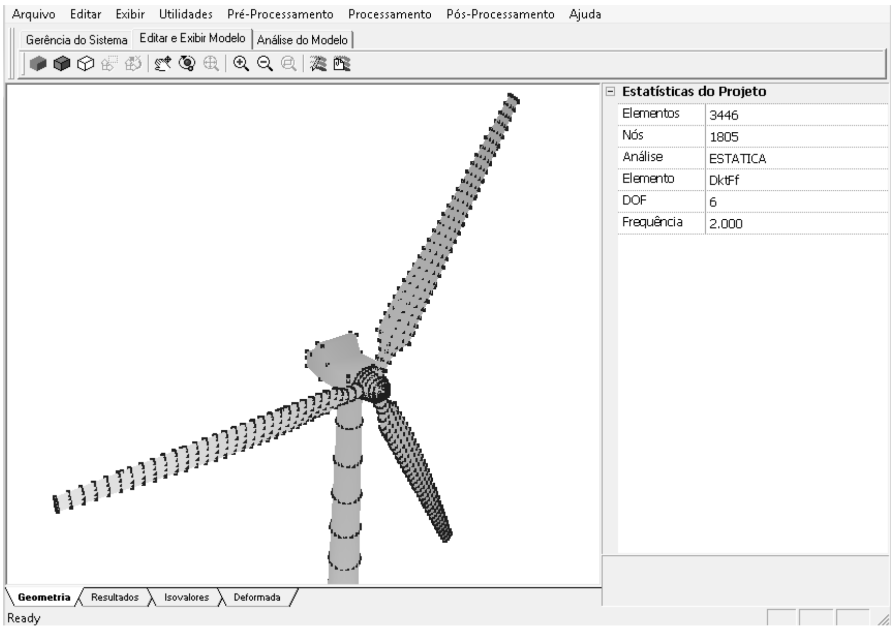

Fig. 1. Full Discretized Model

The tool provides all drawing capabilities necessary to incorporate remaining Finite Element input data information (Elasticity modulus, Poisson ratio, Thickness, etc), the OpenGL library was used in order to implement the drawing routines and to assign structural analysis data.OpenGL is a cross-language API for writing applications that produce 2D and 3D computer Graphics [1]. The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives [2-3]. In Fig 2, a piece of source code to draw the triangle finite element is visualized. The parametric tool use OpenGL when the user selects the menu option “Pre-Processing -> Parametric Modeling”, so that this corresponds to execute the instructions in the source code as shown in Fig. 2.

```
void DrawOpenGLFems()
{

    int i, Vi;
    float length;

    // Draw the faces using an index to the vertex array
    glBegin(GL_TRIANGLES);
    for(i=0;i<=Glgph.NumOfFems-1;i++)
    {

        //Vertex One
        Vi = Glgph.Fems[i].NodeA-1;
        glColor3f(Glgph.Fems[i].ColorRa, Glgph.Fems[i].ColorGa,
        Glgph.Fems[i].ColorBa);

        // Length
        length = (float)sqrt(pow(Glgph.Nodes[Vi].Cx,2) +
            pow(Glgph.Nodes[Vi].Cz,2) +
            pow(Glgph.Nodes[Vi].Cy,2));

        if(length == 0.0f)
            length = 1.0f;

        glNormal3f(Glgph.Nodes[Vi].Cx / length,
            Glgph.Nodes[Vi].Cz / length,
            Glgph.Nodes[Vi].Cy / length);


        glVertex3f(Glgph.Nodes[Vi].Cx, Glgph.Nodes[Vi].Cz,
        Glgph.Nodes[Vi].Cy);


        ...continues to next vertex


    }
    glEnd();

}
```

Fig. 2. Function to draw the triangle finite element

In Fig 3, is shown the source code of the function to draw the nodes of the finite element.

```
void DrawOpenGLNodes()
{
    int i;

    glBegin(GL_POINTS);
    for(i=0;i<=Glgph.NumOfNodes-1;i++)
    {

        glColor3f(0.10f,0.10f,0.10f);
        glVertex3f(Glgph.Nodes[i].Cx, Glgph.Nodes[i].Cz,
        Glgph.Nodes[i].Cy);

    }
    glEnd();

}
```

Fig. 3. Function to draw nodes of the finite element

When the file data with coordinates nodes, elements connections is opened, the mesh of the eolic blade in environmentis automatically generated, producing a graphical representation depicted in Fig. 4.

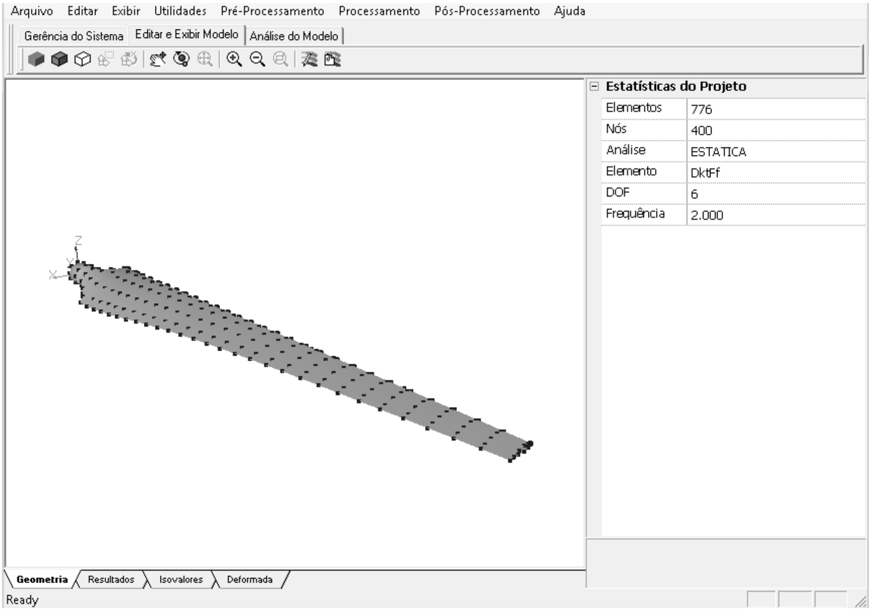

Fig. 4. Discretized Blade Model

The next step in pre-processing is related to the definition of structural element properties. This procedure can be done in specifics dialogsfor the elements. All material propertiesare setting by user and applied to the model. For the nodes, the wind load and boundary conditions (that represent the fixed position for the eolic blade) must be assigned. This can be done in environment dialog box, where the all six degrees in all nodes of the model can be accessed and one by one conveniently prescribed by user, Fig. 5.

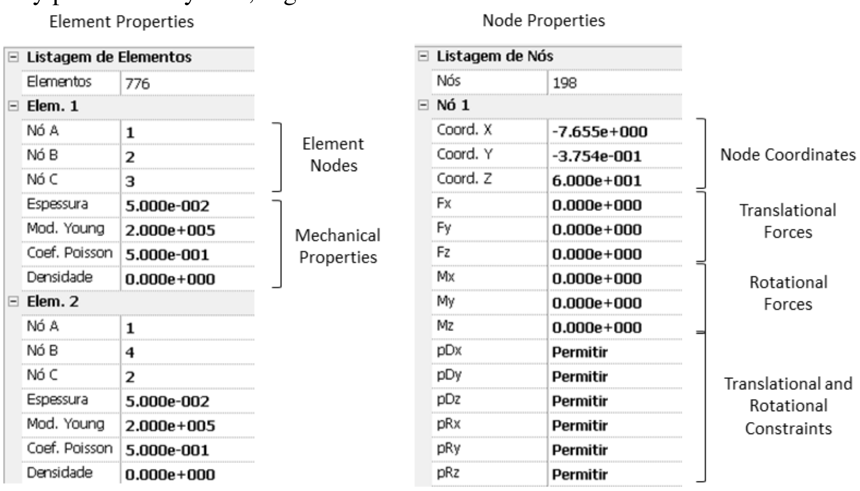

Fig. 5. Element and Node Properties

The final representation of the model, with the wind load and nodal constraints applied can be visualized in Fig. 6. In this stage the model is done for analysis.

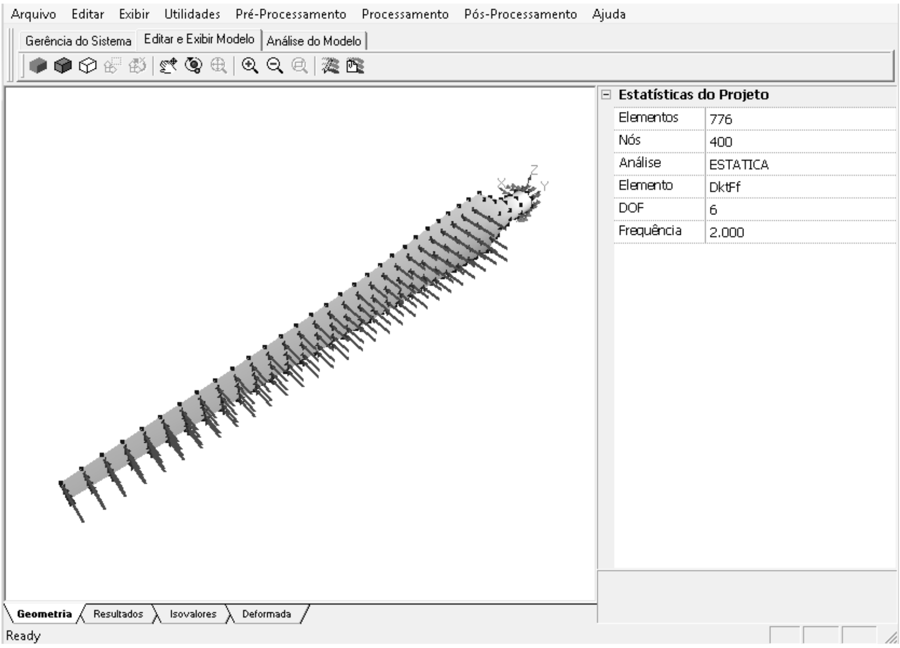

Fig. 6. Wind Load and Fixing Conditions for the Eolic blade

3Processing

The processing is stage of the analysis in which main calculations (such as elemental stiffness matrix evaluation, structural stiffness matrix assembling, nodal equivalent force vector evaluation, algebraic system solution, etc) are done. In present paper, the structural eolic blade problem is analyzed by superposition of bending and membrane effects using Finite Element Method (FEM). For membrane effects is used Free Formulation (FF) finite element originally developed by [7]. The main characteristics of this element are triangular flat geometry, three degrees of freedom (DOF) by node (two displacements on plane and one drilling rotation perpendicular to the plane) located at each triangle vertex. For bending effect it is used the flat triangular element DKT (discrete Kirchhoff Theory), that have three DOF (one transverse displacement and two slopes) by node located at each triangle vertex. This element was developed to deal with thin bending plate problems and its formulation has been thoroughly discussed [4-6]. The eighteen DOF of the wind turbine tower analysis element (bending + membrane problems) are shown in Fig. 7.

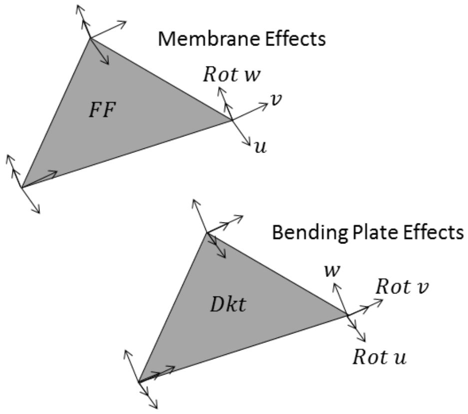

Fig. 7. DKT and FF Finite Elements Association

The analysis is done using the Saproms.dll library, in which have functions and classes implemented in oriented object language C++. For sake of conciseness, only the principal classes and functions of the Saproms.dll are described in Fig. 8.

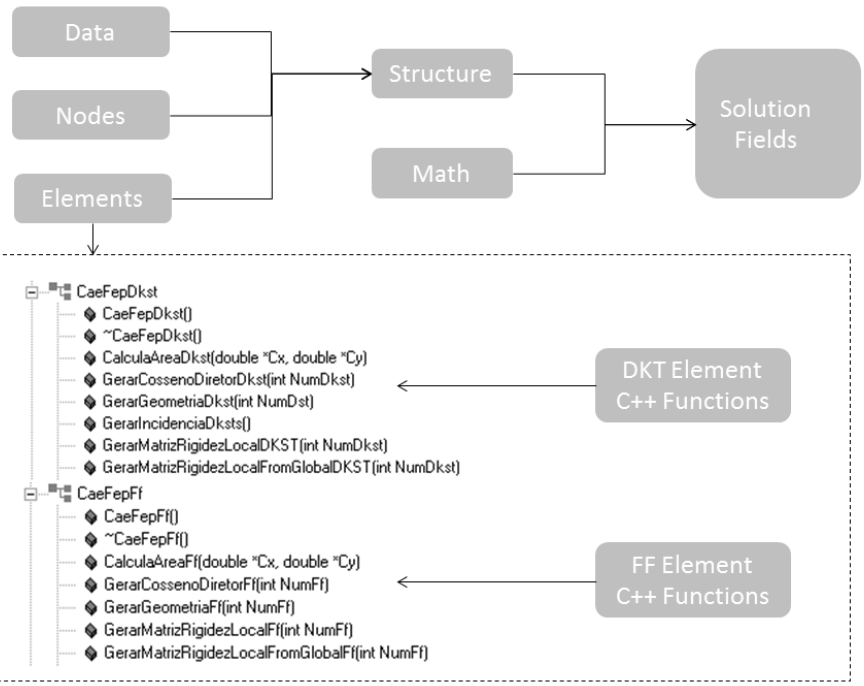

Fig. 8. Classes and Functions

The main function that uses methods and objects from Saproms.dll is "solver". This function works in two steps: in the first the objects and functions in

Saproms.exe collect the input data assigned in pre-processing stage, and store them in vectors. In the second step, these vectors are processed by objects and methods from Saproms.dll to perform the structural calculations.From de point of view of mathematics, the “solver” function for the static analysis uses the principle of total strain energy. The governing equation of the problem is given by.

$$[K]\{D\} = \{P\} \quad (1)$$

Where$[K] = \sum([K_{ff}] + [K_{dkt}])$, is the stiffness matrix of the structure, and$\{D\}$,$\{P\}$ are the vectors of displacement and nodal forces.

The equivalent nodal force vector $\{P\}$is obtained from the external work done by the wind loads is expressed by.

$$T_e = \int_A g(x,y)\, w(x,y) dA \quad (2)$$

Where $w(x,y)$, $g(x,y)$are the displacement and the wind loading in the element, A is the area of the element.

The equivalent nodal force vector is equal to the vector derived from the work of external loads compared to degrees of freedom, so that the establishment of interpolations to $w(x,y)$ and $g(x,y)$along the area of the element is necessary. Assuming a linear variation in Fig. 9.

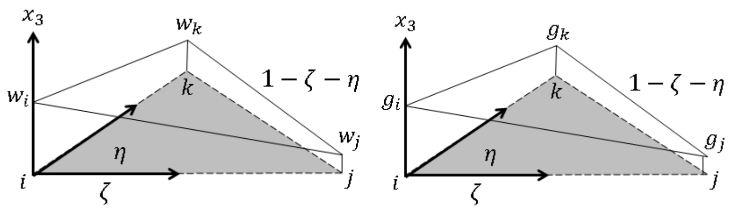

Fig. 9. Interpolating for $w(x,y)$ and $g(x,y)$

$$w = w_i(1-\zeta-\eta) + w_j\zeta + w_k\eta \quad (3)$$

$$g = g_i(1-\zeta-\eta) + g_j\zeta + g_k\eta \quad (4)$$

Substituting (3) and (4) in (2),

$$T_e = \int_A \left(g_i(1-\zeta-\eta) + g_j\zeta + g_k\eta\right)(w_i(1-\zeta-\eta) + w_j\zeta + w_k\eta) dA \quad (5)$$

By minimizing the potential energy due to external loads,

$$\begin{pmatrix} F_i \\ F_j \\ F_k \end{pmatrix} = \begin{pmatrix} \frac{\partial T_e}{\partial w_i} \\ \frac{\partial T_e}{\partial w_j} \\ \frac{\partial T_e}{\partial w_k} \end{pmatrix} = \int_A \begin{pmatrix} (1-\zeta-\eta)^2 & (1-\zeta-\eta)\varsigma & (1-\zeta-\eta)\eta \\ (1-\zeta-\eta)\varsigma & \zeta^2 & \zeta.\eta \\ (1-\zeta-\eta)\eta & \zeta.\eta & \eta^2 \end{pmatrix} dA \begin{pmatrix} g_i \\ g_j \\ g_k \end{pmatrix} \quad (6)$$

After calculating the integral we have the vector of nodal loads that can be given by, see Fig. 10.

$$\begin{pmatrix} F_i \\ F_j \\ F_k \end{pmatrix} = \underset{\sim}{T} \begin{pmatrix} g_i \\ g_j \\ g_k \end{pmatrix} \qquad \text{Where} \quad \underset{\sim}{T} = \frac{A}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \quad (7)$$

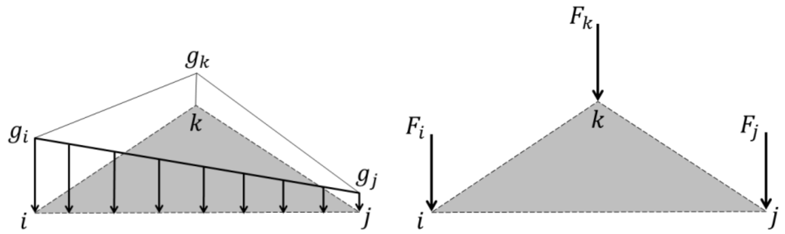

Fig. 10. Wind loadsConvertedto Nodal Equivalent Forces.

Another formulation to the equivalent nodal force vector was proposed by [8], called pseudo-consistent equivalent nodal force vector, where rotational load degrees are calculated considering the external work done by the wind loads expressed by.

$$T_e = \underset{\sim}{u}^T \underset{\sim}{S}^T 2A \int_0^1 \int_0^{1-\eta} \begin{bmatrix} (1-\zeta-\eta) & \zeta & \eta \\ (1-\zeta-\eta)\zeta & \zeta^2 & \zeta\eta \\ (1-\zeta-\eta)\eta & \zeta\eta & \eta^2 \\ (1-\zeta-\eta)\zeta\eta & \zeta^2\eta & \zeta\eta^2 \\ (1-\zeta-\eta)\zeta^2 & \zeta^3 & \zeta^2\eta \\ (1-\zeta-\eta)\eta^2 & \zeta\eta^2 & \eta^3 \\ (1-\zeta-\eta)\zeta^2\eta & \zeta^3\eta & \zeta^2\eta^2 \\ (1-\zeta-\eta)\zeta\eta^2 & \zeta^2\eta^2 & \zeta\eta^3 \\ (1-\zeta-\eta)\zeta^3 & \zeta^4 & \zeta^3\eta \\ (1-\zeta-\eta)\eta^3 & \zeta\eta^3 & \eta^4 \end{bmatrix} d\eta d\zeta \, \underset{\sim}{g} \quad (8)$$

Again, after calculating the integral we have the vector of nodal loads that can be given by (see Fig. 11).

$$\begin{pmatrix} F_i \\ F_j \\ F_k \end{pmatrix} = \underset{\sim}{S}^T \underset{\sim}{T} \begin{pmatrix} g_i \\ g_j \\ g_k \end{pmatrix} \qquad \text{Where} \quad \underset{\sim}{S} = \underset{\sim}{G}^{-1} \underset{\sim}{Q} \tag{9}$$

The $\underset{\sim}{T}$ is

$$\underset{\sim}{T} = \frac{A}{180} \begin{bmatrix} 60 & 60 & 60 \\ 12 & 30 & 12 \\ 12 & 12 & 30 \\ 3 & 6 & 6 \\ 6 & 18 & 6 \\ 6 & 6 & 18 \\ 1 & 3 & 2 \\ 1 & 2 & 3 \\ 3 & 12 & 3 \\ 3 & 3 & 12 \end{bmatrix} \tag{10}$$

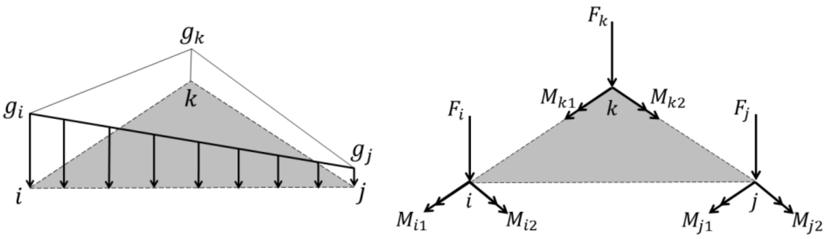

Fig. 11. Wind loadsConvertedtoPseudo-ConsistentNodal Equivalent Forces.

The $\underset{\sim}{G}^{-1}$ and $\underset{\sim}{Q}$ in equation (9) are matrices given in (11) and (12). The explicit values of (11) can be obtained from [9].

$$\underset{\sim}{Q} = \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} & Q_{14} & Q_{15} & Q_{16} & Q_{17} & Q_{18} & Q_{19} \\ Q_{21} & Q_{22} & Q_{23} & Q_{24} & Q_{25} & Q_{26} & Q_{27} & Q_{28} & Q_{29} \\ Q_{31} & Q_{32} & Q_{33} & Q_{34} & Q_{35} & Q_{36} & Q_{37} & Q_{38} & Q_{39} \\ Q_{41} & Q_{42} & Q_{43} & Q_{44} & Q_{45} & Q_{46} & Q_{47} & Q_{48} & Q_{49} \\ Q_{51} & Q_{52} & Q_{53} & Q_{54} & Q_{55} & Q_{56} & Q_{57} & Q_{58} & Q_{59} \\ Q_{61} & Q_{62} & Q_{63} & Q_{64} & Q_{65} & Q_{66} & Q_{67} & Q_{68} & Q_{69} \\ Q_{71} & Q_{72} & Q_{73} & Q_{74} & Q_{75} & Q_{76} & Q_{77} & Q_{78} & Q_{79} \\ Q_{81} & Q_{82} & Q_{83} & Q_{84} & Q_{85} & Q_{86} & Q_{87} & Q_{88} & Q_{89} \\ Q_{91} & Q_{92} & Q_{93} & Q_{94} & Q_{95} & Q_{96} & Q_{97} & Q_{98} & Q_{99} \\ Q_{10,1} & Q_{10,2} & Q_{10,3} & Q_{10,4} & Q_{10,5} & Q_{10,6} & Q_{10,7} & Q_{10,8} & Q_{10,9} \end{bmatrix} \tag{11}$$

$$\underset{\sim}{G}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -11/2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 9 & -9/2 \\ -11/2 & 0 & 1 & 0 & 0 & 0 & -9/12 & 9 & 0 & 0 \\ 18 & 0 & 0 & 27 & -9/2 & -9/2 & 9/2 & -45/2 & -45/2 & 9/2 \\ 9 & -9/2 & 0 & 0 & 0 & 0 & 0 & 0 & -45/2 & 18 \\ 9 & 0 & -9/2 & 0 & 0 & 0 & 18 & -45/2 & 0 & 0 \\ -27/2 & 0 & 0 & -27 & 27/2 & 0 & 0 & 27/2 & 27 & -27/2 \\ -27/2 & 0 & 0 & -27 & 0 & 27/2 & -27/2 & 27 & 27/2 & 0 \\ -9/2 & 9/2 & 0 & 0 & 0 & 0 & 0 & 0 & 27/2 & -27/2 \\ -9/2 & 0 & 9/2 & 0 & 0 & 0 & -27/2 & 27/2 & 0 & 0 \end{bmatrix} \quad (12)$$

4Post-Processing

After discussing the mathematical aspects of the wind load model.This section finally shows the Post-processing results. In user-friendly environment the displacements results can be accessed by graphical outputs as viewed in Fig. 12.

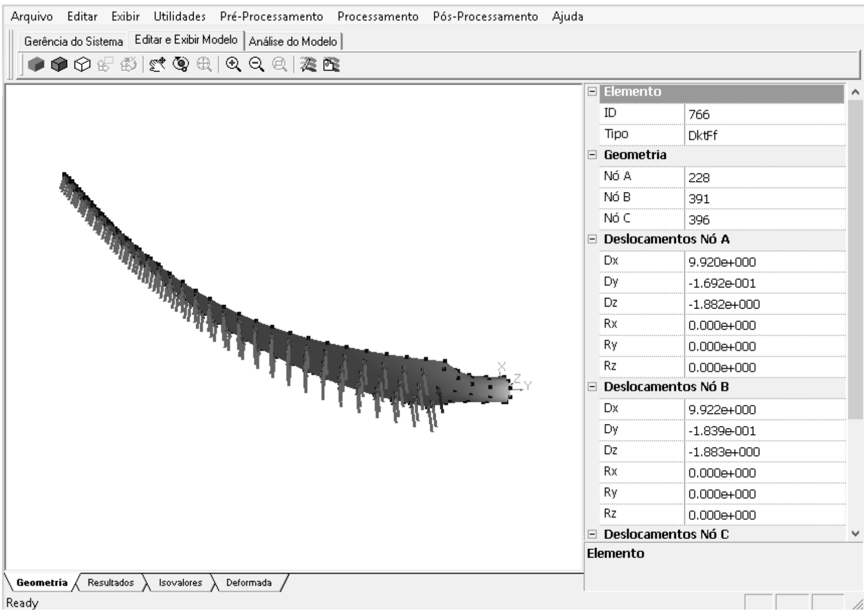

Fig. 12. Displacementresults

The graphical results for the displacement variation on the length of the blade are shown in Fig. 13. The mechanical properties and other input data considered for the model analysis are presented in Fig. 5. The load for the wind pressure in each node in Xdirection on surface of the eolic blade is $1.000e - 02$ N.

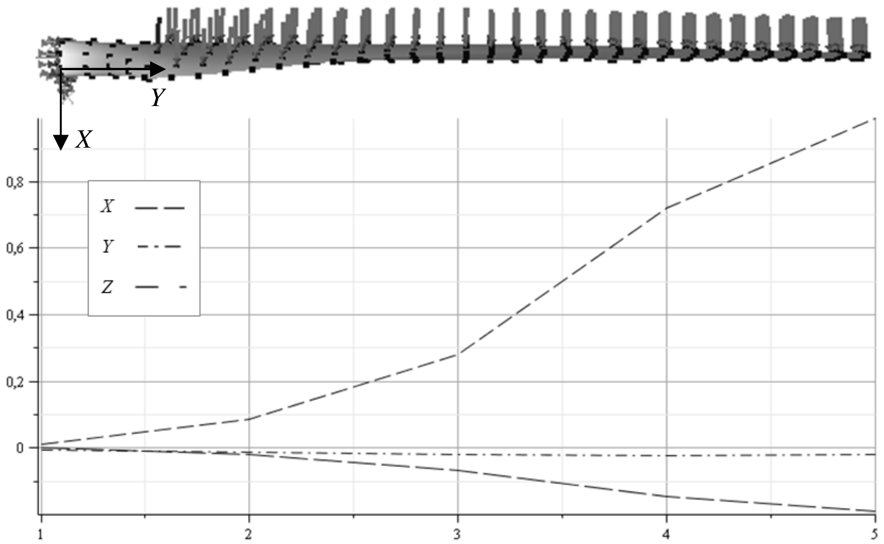

Fig. 13. Displacementresults

5Conclusion

In this paper a tool for analysis of eolic blades was presented. The main attractive feature of this structural analysis tool is the DLL program for the processing stage implemented in C++ language for the FF and DKT finite elements used to discretize the eolic blade.In addition a user-friendly environment was implemented using OpenGL library to provide the graphical construction for geometry, mesh orientation, and other requirements of the finite element model.

References

[1] Foley et al.; Computer Graphics - Principles and Practice with C; Addison-Wesley.

[2] Woo, Neder& David.; OpenGL Programming Guide; Addison-Wesley.

[3] Hearn & Baker; Computer Graphics with OpenGL; Prentice Hall.

[4] Batoz JL, Bathe KJ, Ho LW. A study of three-node triangular plate bending elements. Int J Numer Meth Engng 1980;15:1771–812.

[5] Batoz JL, Lardeur P. A discrete shear triangular nine d.o.f. element for the analysis of thick to very thin plates. Int J Numer Meth Engng 1989;28:533–560

[6] Batoz JL, Dhatt GS. Mod_elisation des Structures par _Elements Finis, 3, Coques, Hermes, Paris, 1992. p. 448-55.

[7] Bergman, P. G.; Felippa, C. A.(1985) " A triangular membrane element with rotational degrees of freedom" Comp. Meths. in Applied Mech. Eng., v.50, p.25-69.

[8] Sydenstricker, R.M; Landau. L., ''Study of some triangular discrete Reissner-Mindlin plate shell elements. Computers and Structures, Volume 78, p. 21–33, 2000.

[9] Viana ,H. R. G. (2008). Análise de placas e vibratória de placasutilizando-se o método de elementosfinitos.Dissertação. Universidade Federal da Paraíba.

[10] Menezes Junior, R. A. "Análise da interação dos efeitosplaca-pórticoemproblemas de edifíciosutilizando-se programaçãoorientadaaoobjetoem C++". PIBIC 2005-2006, Universidade Federal da Paraíba, João Pessoa, PB, Brasil.