RESEARCH ARTICLE OPEN ACCESS

Mathematical Model and Programming in VBA Excel for Package Calculation

João Daniel Reis Lessa[1], Paulo Fabiano Reis Lessa[2], Pedro Américo Almeida Magalhães Junior[3], Henrique de Vilhena Guimarães[4]
[1]*Department of Mechanical Engineering, PUC Minas University, Brazil*
[2]*Department of Mechanical Engineering, PUC Minas University, Brazil*
[3]*Department of Mechanical Engineering, PUC Minas University, Brazil*
[4]*Department of Mechanical Engineering, PUC Minas University, Brazil*

ABSTRACT
The industrial logistics is a fundamental pillar for the survival of companies in the actual increasingly competitive market. It is not exclusively about controlling the flow of external material between suppliers and the company, but for developing a detailed study of how to plan, control, handle and package those materials as well. Logistics activities must ensure the maximum efficiency in using corporate resources once they do not add value to the final product. The creation of a logistic plan, for each piece of the company's production, has to adapt the demand parameters, seasonal or not, in the timeline. Thus, the definition of packaging (transportation and consumption) must adjust in accordance with the demand, in order to allow the logistic planning to work, constantly, with order of economy batches. The packaging calculation for each part in every demand can become well complicated due to the large amount of parts in the production process. Automating the calculation process for choosing the right package for each piece is an effective method in logistics planning. This article will expose a simple and practical mathematical model for automating the packaging calculation and a logic program, created in Visual Basic language in the Excel software, used for creating graphic designs that show how the packages are being filled.
***Keywords*-**Logistic, packaging, package, Excel file, dimensioning.

I. INTRODUCTION

The society daily life was completely changed by the use of computers through the time whether in professional or in the social interactions. It is possible to observe the companies' dependence on technology, which all information is managed through many controlling software, and all data is stored in dedicated servers.

Just like every other departments in a big company, the industrial logistic is increasingly using technology in its management activities, and tasks execution. Applications and computer software such as Microsoft Excel have become essential in logistics activities of data optimization, and decision making.

Analyzing the industrial logistics role at the overall company decisions increasing efficiency, in transportation costs, handling and storage of materials, it is possible to have a special concern over inventory levels. According to Correa and Gianesi (2011), the principles of materials management say that it is necessary to determine the purchasing and production lots through the balance between the costs of maintaining inventories and fixed costs related to obtaining the lot (costs with the preparation of equipment in case of produced items, and costs to process purchase orders, in case of purchased items) [1]. An efficient manner to ensure that the production and purchase of materials are in accord to demand is determining an economic lot of order that needs the guarantee of choosing a suitable packaging.

The choice and dimensioning of packages in logistics activities do not have methodologies or specific techniques available in the current literature collection. According to Moura (1998), inventory control can be optimized by grouping and packaging the volume of items in plastic boxes and the reduction of human error in repetitive activities [2].

Thus, the purpose of this article is to use programming tools, offered by Excel, in choosing a suitable package for each managed item, and eliminating a repetitive activity in logistics planning as well.

II. JUSTIFICATION

Observing that the biggest companies are currently focused in production of goods, it is clearly possible to check the role of industrial logistics in the optimization of resources spent with infrastructure and inventory planning. Properly define the storage packaging components for the

production lines of the company is only one of many activities related to logistics. However, when thinking about thousands of different parts and the demand for each one, this activity will surely reflect directly in the production chain, either by excess amount of parts in the inventory or lack of raw materials at the right time for production.

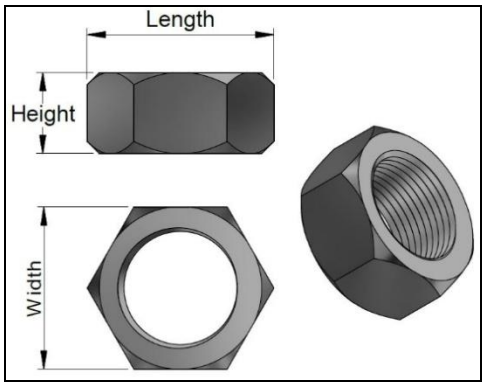
The idea of creating an automated system for the study of dimensioning, and package selection arose from the difficulty to define, for distinct parts, the correct amount of parts that could be allocated within different types of standard packaging, also allowing the best choice of packaging related to the part demand.

III. CALCULATION METHODOLOGY

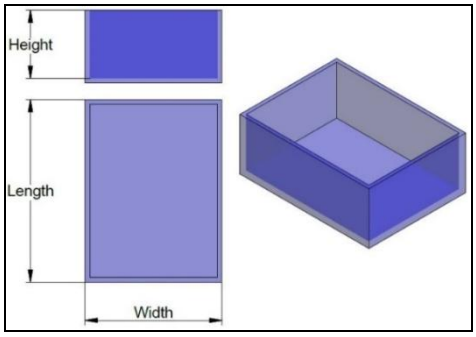
3.1. Concepts and Parameters

The method concepts used for the calculation is based on the assumption that there are already defined standards packages with known dimensions, and also parts with known dimensions. Therefore, it is desired to determine the maximum possible number of parts to be allocated inside each package available. After the dimensioning of each registered package be properly defined, it is possible to select the one, which best meets the demand of the part and logistic parameters adopted by the company, for example: economic lot of request, days of inventory, storage capacity, etc.

The main concept used to calculate the maximum number of pieces that can effectively be placed in an aleatory package, is based on transforming the actual format of the part into a parallelepiped which its dimensions are proportional to the height, width and length of the part, as shown in Figure 1. Then, the dimensions of this equivalent parallelepiped are compared with the internal dimensions of each package (Figure 2).

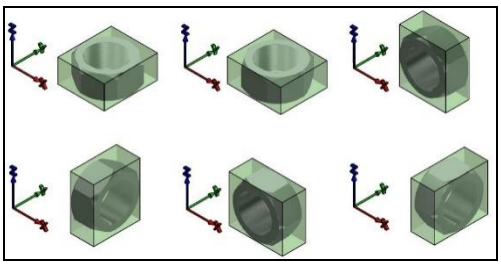


1. Figure: Main Dimensions of Part



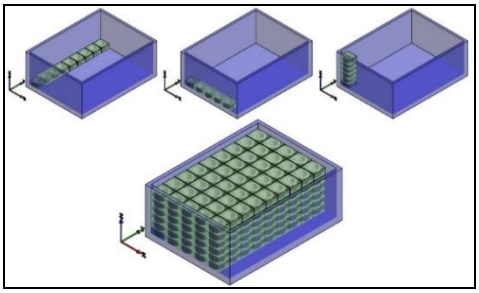
2. Figure: Main Dimensions of Package

In order to determine the best way to allocate the item inside the package, it is necessary to consider all possible positions that it can be placed, as shown in Figure 3. It is important to highlight that only the main dimensions of the parallelepiped are considered to calculate, so it may imply in some limitations for this methodology that will be discussed in the next topics.



3. Figure: Possible Positions for Each Item

Based on the parameters discussed above, it is possible to better understand how the calculation method will set the quantity of pieces for a particular item, and how it will fill a particular package. For each axis (x, y, z), as shown in Figure 3, is calculated the maximum number of parts that can be allocated inside the package. The quantity of parts allocated depends directly on the internal dimensions of the package (Length, Height and Width). Then, it is possible to calculate the total number of parts on it. The Figure 2 shows exactly how this calculation is done.



4. Figure: Filling Steps for Each Package

3.2. Formulas

After the main concepts and all parameters involved in the calculation method be defined, it is possible to get into the calculation steps and the formulas used. It is valid to remember that all commands shown in this article were executed in the Excel software, the logic can be adapted for any programming language as well, for example: MatLab, ScyLab and so on.

In order to ease the next steps understanding, it will be considered the letters H, W and L as Height, Width and Length of internal dimensions of the package, respectively. In addition, h, w and l will be considered, respectively, as height, width and length of the part that it is desired to be allocated inside the package.

3.2.1. Definition of Parts Quantity inside Package

In order to quantify the amount of parts inside the package by using the Excel software, it is considered each one of the six possible positions of the part inside the package (Figure 2), as shown in Figure 3. Furthermore, it is needed to create a table similar to Table 1, which is composed by cells filled with the composition of the three internal dimensions of the package (H, W and L) and the three dimensions of the part (h, w and l). Then, it is generated six different results. This operation is considered the first step for determining the amount of parts that is possible to place inside the package.

	A	B	C	D	E	F
1		H	W	L		
2	h	H/h	-	-		
3	w	H/w	W/w	-		
4	l	H/l	W/l	L/l	=ROUNDDOWN(B2;0)*ROUNDDOWN(C3;0)*ROUNDDOWN(D4;0)	1st Position
5	h	-	W/h	L/h	=ROUNDDOWN(B3;0)*ROUNDDOWN(C4;0)*ROUNDDOWN(D5;0)	2nd Position
6	w	-	-	L/w	=ROUNDDOWN(B4;0)*ROUNDDOWN(C5;0)*ROUNDDOWN(D6;0)	3rd Position
7	l	H/l	-	-		
8	w	H/w	W/w	-		
9	h	H/h	W/h	L/h	=ROUNDDOWN(B7;0)*ROUNDDOWN(C8;0)*ROUNDDOWN(D9;0)	4th Position
10	l	-	W/l	L/l	=ROUNDDOWN(B8;0)*ROUNDDOWN(C9;0)*ROUNDDOWN(D10;0)	5th Position
11	w	-	-	L/w	=ROUNDDOWN(B9;0)*ROUNDDOWN(C10;0)*ROUNDDOWN(D11;0)	6th Position

1. Table: Calculation of Quantity of Parts

As shown at the table above, the total amount of parts within the package is determined by multiplying the three quotients obtained from the first step by the part dimensions in each different position.

It is worth noting that in the E column, each division is rounded down, by the formula ROUNDDOWN, for generating an integer number. The rounding is necessary because the quantified part must not be split or exceed the package size.

To better understanding of the methodology described previously, the Table 2 shows an example of a calculation realized from dimensions pre-defined by the user.

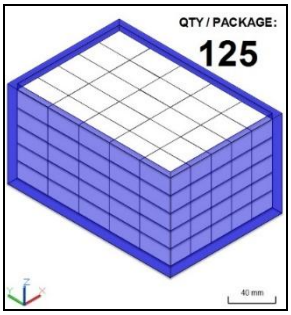
	H	W	L		
h	5,33	-	10		
w	4	5	7,5		
l	2,67	3,33	5	125	1st Position
h	-	6,67	10	120	2nd Position
w	-	-	7,5	84	3rd Position
l	2,67	-	-		
w	4	5	-		
h	5,33	6,67	10	100	4th Position
l	-	3,33	5	120	5th Position
w	-	-	7,5	105	6th Position

H	80 mm
W	100 mm
L	150 mm

h	15 mm
w	20 mm
l	30 mm

2. Table: Calculation of Quantity of Parts

The simulation shows the internal dimensions of a package randomly chosen, and the external dimensions of a part to be quantified inside the package, which were pre-defined in the spreadsheet. Note that six results are obtained for the six possible positions of the part within the package. Therefore, the position which fits the largest amount of parts can be allocated is the first one - total of 125 parts allocated inside the package. The figure 5 shows how the filling of the package looks like, with the part in the first position.

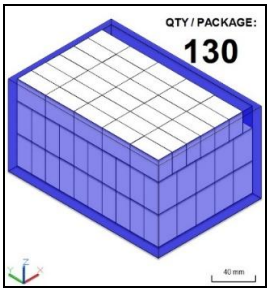


5. Figure: Package Filled with one Position of the Part

To make the calculation more reliable, comparing with the actual process of filling the package, others parameters can be inserted, as such:

- Gap between parts: It is used when the parts fit too close one of another within the package, making the process of filling the package extremely difficult.
- Utilization of empty space: When the calculation for each position of the part is done, it can happen that some certain space inside the package stays empty. This space may be filled with the same part in a different position besides of the position firstly considered. This step obtain the three dimensions of the empty space then repeat the same calculation process done previously, in order to obtain the complementary quantity of parts that can be allocated within the package. The Figure 6 shows how the utilization of the package looks like, for the same example shown in the Table 2. The quantity of 125 parts is increased to 130

parts due to the sum of parts allocated in a different position.



6. Figure: Package Filled with the Part in Two Different Positions

- Weight Capacity of the Package: Sometimes the amount of parts calculated by this methodology can exceed the maximum weight supported by the package. In those cases, it is necessary to create a specific function that restricts that the total weight force of the parts does not exceed the weight limit of the package. Because of it, it is determined the largest amount of parts based on the maximum weight supported by the package.

3.2.2. How to determine the best package

The process to determine the maximum number of parts that can be allocated in a package randomly chosen does not define the best package for that determined part. This process only pops up the maximum amount of parts for a pre-defined package. However, in order to define the best package to fit the part, it is necessary to analyze some logistics patterns that precedes in this topic.

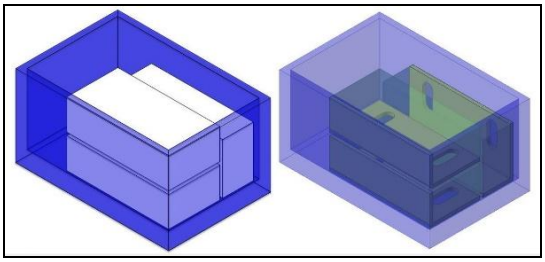
Generically speaking, the big companies have many types of package with standard dimensions. Beyond the available packages, each company has its own methodology of inventory managing, and that defines the maximum weight of the packages, the inventory level, the request of an economic lot, the number of parts desired in the inventory, etc. Then, to define the best package it is necessary to:

- Calculate the best way to fill the package for that specific part. On this step, it is only necessary to repeat the calculation done before for determining the largest amount of parts allocated for each package available on the logistics scope. Thus, it is only needed to change the package internal dimensions in the software, such as: Height, width, length, and the maximum weight supported by the box.
- Determine the minimum amount of parts that must be allocated for each package. After defining this variable, the software is going to be able to point the best package that attends all requirements inserted by the user.

3.3. Limitations

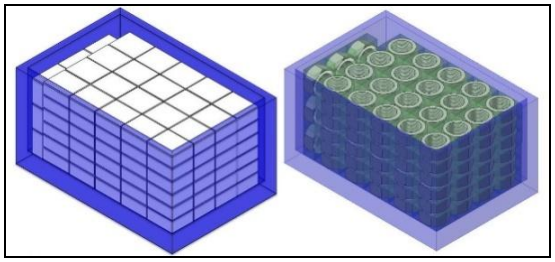
Even with all advantages provided by the usage of this methodology, there are some limitations related with its application.

As previously studied, the calculation method transforms all geometric formats into parallelepipeds, simplifying the dimensions of the part to its maximum value of length, width, and height. Because of it, there is the possibility of the parts fit in a different position within the package besides from the other six possible positions calculated by this methodology, and then, in this new position, the maximum number of parts allocated can be larger than the result shown by the method. Besides that, frequently it is desirable to scale out malleable items like, electric cables, rubber hoses, adhesives, etc. For these parts, the spreadsheet will not consider the compacted dimensions automatically, but the rigid ones, making the methodology not reliable for these parts. The Figure 7 shows an example of one limitation that would be necessary a deep calculation to define the maximum number of parts allocated, or the best package through this calculation method.



7. Figure: Limitation Example for the Calculations Methodology

Briefly, these simplifications made by the calculation process are perfectly applicable for items which its external volume (relative to external dimensions Height, Width, and Length) does not exceed in 15% the actual volume of the workpiece (except for rigid parts with internal spaces like tubes, nuts, washers, etc.). The Figure 8 shows a viable example calculated by the method.



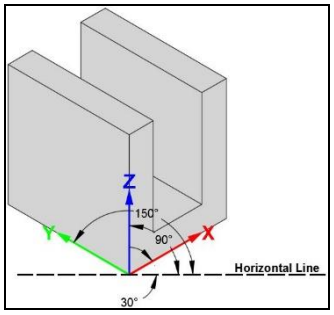
8. Figure: Viability Example of the Calculation Methodology

IV. DRAWING METHODOLOGY

4.1. Concepts and input information to plot the graphic

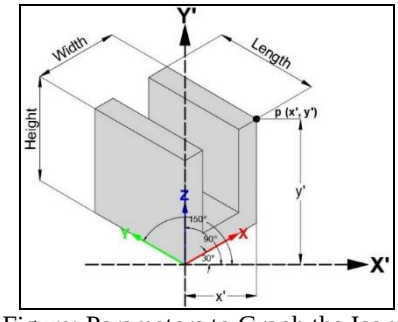
The graphic will be constructed relating the external and internal dimensions of the package with the external dimensions of all parts quantified by the software, in isometric view. The result will be an isometric drawing of the package that shows the position of the parts allocated on it, as shown in the Figures 5 and 6.

The isometric perspective maintains the same proportions for the length, width and height of the object represented which makes this procedure ideal for simplifying the functions, and program routines to construct the drawing. The axles *X*, *Y* e *Z* have the angles of 30°, 150º and 90° respectively, in relation to the field of view, as shown in Figure 9.



9. Figure: Isometric Perspective

The Excel desktop only has two dimensions (*X'* and *Y'*). Therefore, to represent the three dimensions (*X*, *Y*, and *Z*) on Excel desktop as shown in the figure above, it is necessary to consider the dimensions of the package and the parts, and yet the sines and cosines of the angles (30°, 150º and 90°) to determine the coordinates (*x'* and *y'*) for each vertex that composes the isometric image. The Figure 10 illustrates the *x'* and *y'* coordinates and how they were obtained.



10. Figure: Parameters to Graph the Isometric Perspective

The first step is to define origin point (*x0'* and *y0'*) in the work area of any tab of Excel. This point (*x0'* and *y0'*) will be the reference for all others coordinates (*x'* and *y'*). It also includes the coordinates of the vertices which compose the external dimensions of the package, the coordinates of the vertices which compose the internal dimensions of package, and the coordinates of the vertices which compose the external dimensions of each part allocated within the package. Consequently, it is created a complete map of the desired graphic.

In order to define the coordinates for a point *'p'*, randomly chosen, it is only needed to associate the parameters of *Height* and *Width* with sines and cosines of the angles of the perspective. Using basic concepts of trigonometry, the coordinate *x'* and *y'* is obtained by:

- *x' = Width * cos(30);*
- *y' = Width * sen(30) + Height.*

This procedure must be repeated for all other vertices of the figure until the complete spatial map is obtained. It is important to highlight some aspects that precede the coordinate definitions.

- Excel works with the positioning of the *Y'* axis, coordinate unlike a common Cartesian plane, because the increase of *y* is vertical from top to bottom. Thus, it is necessary to consider such a requirement in the definition of coordinates within the VBA desktop;
- The absolute values of the dimensions of the package and the items allocated within the package do not fit the area available in Excel tabs and, in addition, the software works with unit of measure in pixels unlike the measures in millimeters used in the calculation method. Thus, it is necessary to create a scale factor to fit the entire drawing to a desired area in the Excel environment.

4.2. VBA Routines

The basic routines for plotting the graph in Excel will use two Excel tabs. The first tab will be used for all calculations to define the amount of parts allocated within the package and all suitable parameters for drawing. The second one will be available to create the package drawing with the filling parts on it.

4.2.1. delete_all_pictures() Routine

The *delete_all_pictures ()* routine has the function to delete all existing figures destined to the exhibition of package drawing in the Excel tab. This routine is fundamental to ensure that the area destined to make the drawing is clear when the main routine is executed. The Figure 11 shows an example of how to create this routine.

```
'Define routine name
Private Sub delete_all_pictures()

    'Define the word "picture" as a Shape
    Dim picture As Shape

    'Delete all shapes less named "locked_picture"
    For Each picture In ActiveSheet.Shapes
        If picture.name <> "locked_picture" Then
            picture.delete
        End If
    Next picture

End Sub
```

11. Figure: delete_all_pictures() Routine

In case that some image needs to be kept in order to compose the graph layout, the programming code has a conditional which all images named *"locked_picture"* will not be deleted.

4.2.2. drawing_plan() Routine

The *drawing_plan()* routine is responsible for creating polygons based on four vertices. In addition, it is possible to set the color and the transparency for each polygon. Integrated into the main routine *main()*, the *drawing_plan()* routine will be responsible for drawing all plans that forms the image of the complete package with all parts allocated within it. The Figure 12 shows the *drawing_plan()* routine with all needed input information.

```
'Define the arguments and the routine name
Private Sub drawing_plan(px1, px2, px3, px4, py1, py2, py3, py4, red, green, blue, transparency)

    'Insert a shape based in four points starting with px1 and py1
    With ActiveSheet.Shapes.BuildFreeform(msoEditingAuto, px1, py1)
        .AddNodes msoSegmentLine, msoEditingAuto, px2, py2
        .AddNodes msoSegmentLine, msoEditingAuto, px3, py3
        .AddNodes msoSegmentLine, msoEditingAuto, px4, py4
        .AddNodes msoSegmentLine, msoEditingAuto, px1, py1
        .ConvertToShape.Select
    End With

    'Define line style
    With Selection.ShapeRange.Line
        .Visible = msoTrue
        .ForeColor.RGB = RGB(0, 0, 0)
        .transparency = 0
        .Visible = msoTrue
        .Weight = 0.25
    End With

    'Define fill style
    With Selection.ShapeRange.Fill
        .Visible = msoTrue
        .ForeColor.RGB = RGB(red, green, blue)
        .transparency = transparency
    End With

End Sub
```

12. Figure: drawing_plan() Routine

4.2.4. main() Routine

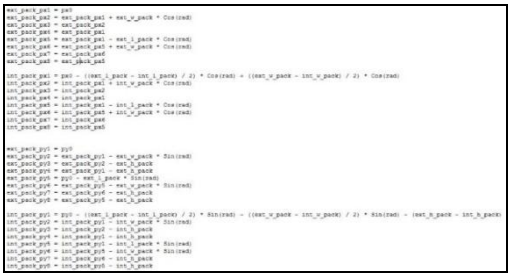
The *main()* routine will execute all necessary commands to make the graph, and for that reason it is the most complex routine.

The first section of this routine is shown in Figure 13 which is defined all input information obtained from the Excel tab named Sheet1. The user previously defined this tab, by using the calculation methodology for package dimensioning.

```
Sub main()

On Error Resume Next

Call delete_all_pictures

rad = 30 * 3.14159265358979 / 180

px0 = Sheets("Sheet1").Cells(1, 2).Value
py0 = Sheets("Sheet1").Cells(2, 2).Value
clearance = Sheets("Sheet1").Cells(3, 2).Value
ext_h_pack = Sheets("Sheet1").Cells(4, 2).Value
ext_w_pack = Sheets("Sheet1").Cells(5, 2).Value
ext_l_pack = Sheets("Sheet1").Cells(6, 2).Value
int_h_pack = Sheets("Sheet1").Cells(7, 2).Value
int_w_pack = Sheets("Sheet1").Cells(8, 2).Value
int_l_pack = Sheets("Sheet1").Cells(9, 2).Value
h_part = Sheets("Sheet1").Cells(10, 2).Value
w_part = Sheets("Sheet1").Cells(11, 2).Value
l_part = Sheets("Sheet1").Cells(12, 2).Value
qty_X_part = Sheets("Sheet1").Cells(13, 2).Value
qty_Y_part = Sheets("Sheet1").Cells(14, 2).Value
qty_Z_part = Sheets("Sheet1").Cells(15, 2).Value
total_qty_part = Sheets("Sheet1").Cells(16, 2).Value
```

13. Figure: Section 1 of 5 of main() Routine

Before making the graph, it is necessary to define which internal and external vertices that will be fixed starting from the origin points (*px0* and *py0*) defined previously. The Figure 14 shows the coordinates definition for the sixteen vertices of the package.



14. Figure: Section 2 of 5 of the main() Routine

After obtaining all the coordinates of the package vertices, Figure 15 shows an execution sequence of the *drawing_plan()* routine. It is responsible for tracing the first plans that will be half of the selected package.

```
Call drawing_plan(ext_pack_px1, ext_pack_px2, ext_pack_px6, ext_pack_px5, _
ext_pack_py1, ext_pack_py2, ext_pack_py6, ext_pack_py5, 0, 0, 255, 0.6)
Call drawing_plan(ext_pack_px2, ext_pack_px3, ext_pack_px7, ext_pack_px6, _
ext_pack_py2, ext_pack_py3, ext_pack_py7, ext_pack_py6, 0, 0, 255, 0.6)
Call drawing_plan(ext_pack_px5, ext_pack_px6, ext_pack_px7, ext_pack_px8, _
ext_pack_py5, ext_pack_py6, ext_pack_py7, ext_pack_py8, 0, 0, 255, 0.6)
Call drawing_plan(int_pack_px1, int_pack_px2, int_pack_px6, int_pack_px5, _
int_pack_py1, int_pack_py2, int_pack_py6, int_pack_py5, 0, 0, 255, 0.6)
Call drawing_plan(int_pack_px2, int_pack_px3, int_pack_px7, int_pack_px6, _
int_pack_py2, int_pack_py3, int_pack_py7, int_pack_py6, 0, 0, 255, 0.6)
Call drawing_plan(int_pack_px5, int_pack_px6, int_pack_px7, int_pack_px8, _
int_pack_py5, int_pack_py6, int_pack_py7, int_pack_py8, 0, 0, 255, 0.6)
Call drawing_plan(int_pack_px3, ext_pack_px3, ext_pack_px7, int_pack_px7, _
int_pack_py3, ext_pack_py3, ext_pack_py7, int_pack_py7, 0, 0, 255, 0.6)
Call drawing_plan(int_pack_px7, ext_pack_px7, ext_pack_px8, int_pack_px8, _
int_pack_py7, ext_pack_py7, ext_pack_py8, int_pack_py8, 0, 0, 255, 0.6)
```

15. Figure: Section 3 of 5 of the main() Routine

For the drawing of all the parts that fill the package does not overlap the package itself in the Excel view after the procedures of Figure 15. The commands responsible for designing all the parts in a sequence of exhibition appropriated based on the methodology that will choose the best position to allocate the parts within the package, as described in

the first part of this article. The Figure 16 shows such commands.

```
counter = 0
px00 = inc_pack_px1
py00 = int_pack_py1 - clearance / 2
For Z = 1 To qty_Z_part
  For Y = 1 To qty_Y_part
    For X = 1 To qty_X_part
      i = qty_X_part - X
      j = qty_Y_part - Y
      k = Z
      counter = counter + 1
      If counter <= total_qty_part Then
        px1 = px00 + i * (w_part + clearance) * Cos(rad) - j * (l_part + clearance) * Cos(rad)
        px2 = px1 + w_part * Cos(rad)
        px3 = px2
        px4 = px1
        px5 = px1 - l_part * Cos(rad)
        px6 = px4 + w_part * Cos(rad)
        px7 = px6
        px8 = px5
        py1 = py00 - i * (w_part + clearance) * Sin(rad) - j * (l_part + clearance) * Sin(rad) - (k - 1) * (h_part + clearance)
        py2 = py1 - w_part * Sin(rad)
        py3 = py2 - h_part
        py4 = py1 - h_part
        py5 = py1 - l_part * Sin(rad)
        py6 = py5 - w_part * Sin(rad)
        py7 = py6 - h_part
        py8 = py5 - h_part
        Call drawing_plan(px1, px2, px3, px4, py1, py2, py3, py4, 255, 255, 255, 0)
        Call drawing_plan(px1, px4, px8, px5, py1, py4, py8, py5, 255, 255, 255, 0)
        Call drawing_plan(px4, px3, px7, px8, py4, py3, py7, py8, 255, 255, 255, 0)
      Else:
        Z = qty_Z_part
        Y = qty_Y_part
        X = qty_X_part
      End If
    Next X
  Next Y
Next Z
```

16. Figure: Section 4 of 5 of the main() Routine

The finite loop created by the sequence of three commands of "For" has the function of ensuring that each piece is drawn until the completion of all dimensions of height, width and length of the package, or until the number of the packing material does not exceed the maximum charge (in this case the counting is done through the "counter").

After the *main()* routine, the last commands responsible for completing the drawing of the packaging will run. The Figure 17 shows the final programming lines of the main routine.

```
Call drawing_plan(int_pack_px1, int_pack_px2, int_pack_px3, int_pack_px4, _
int_pack_py1, int_pack_py2, int_pack_py3, int_pack_py4, 0, 0, 255, 0.6)
Call drawing_plan(int_pack_px1, int_pack_px4, int_pack_px8, int_pack_px5, _
int_pack_py1, int_pack_py4, int_pack_py8, int_pack_py5, 0, 0, 255, 0.6)
Call drawing_plan(ext_pack_px1, ext_pack_px2, ext_pack_px3, ext_pack_px4, _
ext_pack_py1, ext_pack_py2, ext_pack_py3, ext_pack_py4, 0, 0, 255, 0.6)
Call drawing_plan(ext_pack_px1, ext_pack_px4, ext_pack_px8, ext_pack_px5, _
ext_pack_py1, ext_pack_py4, ext_pack_py8, ext_pack_py5, 0, 0, 255, 0.6)
Call drawing_plan(ext_pack_px4, int_pack_px4, int_pack_px3, ext_pack_px3, _
ext_pack_py4, int_pack_py4, int_pack_py3, ext_pack_py3, 0, 0, 255, 0.6)
Call drawing_plan(ext_pack_px4, int_pack_px4, int_pack_px8, ext_pack_px8, _
ext_pack_py4, int_pack_py4, int_pack_py8, ext_pack_py8, 0, 0, 255, 0.6)

End Sub
```

17. Figure: Section 5 of 5 of the main() Routine

V. RESULTS

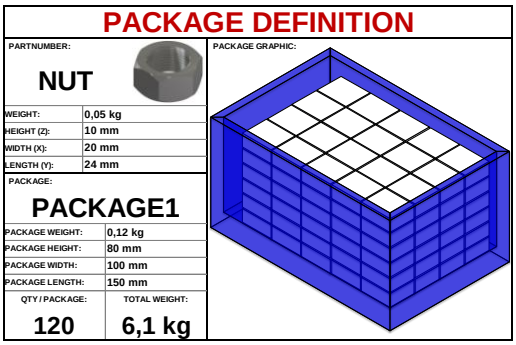
All the methods discussed in this article were created with two main objectives: to choose the best package for a particular item and create a graph with the chosen package illustrating the positioning of the parts within of it. As a result, it will be considered examples of input parameters for the implementation of the methods discussed.

The Table 3 shows a list of packages and their respective characteristics available for selection within the spreadsheet.

NAME	WEIGHT [kg]	MAX_WEIGHT [kg]	EXT_HEIGHT [mm]	EXT_WIDTH [mm]	EXT_LENGTH [mm]	INT_HEIGHT [mm]	INT_WIDTH [mm]	INT_LENGTH [mm]
PACKAGE1	0,12	15	80	100	150	70	90	140
PACKAGE2	0,20	15	80	150	200	70	140	190
PACKAGE3	0,57	15	150	200	300	130	190	280
PACKAGE4	1,08	15	150	300	400	130	280	370

3. Table: List of Package

The Figure 18 shows the result, based on the procedures of this article. The external dimensions of the nut were illustrated, and were stipulated a demand of 80 parts inside the package, which resulted in the choice of PACKAGE1 with 120 parts positioned according to the graph built automatically.



18. Figure: Result in the Software

VI. CONCLUSION

The analysis done to obtain the described results showed a simple and functional possibility of automating the job of choosing the best package that will compose the logistics plan for each piece managed. By using the described methodology, it is possible to document or display the work of choosing the best package, and streamline the work of characterization of the logistics items.

Although the methods presented are not applicable to special geometries items, which means that, when transformed into parallelepipeds does not exceed 15% of the actual volume of the part. Thus, it can be understood that such procedures are applicable to several common items in different segments of production, such as screws, nuts, washers, blocks, among others.

ACKNOWLEDGEMENTS

Studies showed an interesting application using VBA basic commands linked to the fundamental principles of geometry and trigonometry in order to obtain a result of exhibition for presentations and work documentation for choosing the package.

In this aspect, the procedures showed the possibilities that a programming language offers to simplify work and repetitive calculations. Thus, through the utilization of the VBA, it was possible to improve the work of the employees letting it much more accurate and efficient.

REFERENCES

[1]. GIANESE, I; CORREA, H. Just-in-Time, MRPII e OPT: um enfoque estratégico. Atlas, São Paulo, 2011.

[2]. MOURA, R. A. Sistemas e Técnicas de Movimentação e Armazenagem de Materiais. São Paulo: Manual de Logística - IMAM. Vol. 1, 1998.