FACTA UNIVERSITATIS
Series: **Mechanical Engineering** Vol. 11, N° 1, 2013, pp. 65 - 74

PROGRESSIVE METHODS AND THE TOOLS USED IN NEW PALLETIZING SOFTWARE *

UDC 621.869

Dušan Kravec, Marián Tolnay, Ondrej Staš, Michal Bachratý

Faculty of Mechanical Engineering, Slovak University of Technology in Bratislava, Slovakia

Abstract. *This article describes the software usable in the automated palletizing process. In addition to the implemented deployment methods, other strong tools such as the two-dimensional drag & drop editor and the three-dimensional preview and editor (virtual reality) are available, too. Editing of new positions for the objects is facing the problem of collisions. This problem is solved by implemented calculations and algorithms. Software output contains coordinates and orientation of each object. This is a closer look inside of it, its functions, the main idea of the program as well as pictures of its GUI (Graphical User Interface).*

Key Words: *Palletizing, Automation, Software, Deployment method*

1. INTRODUCTION

Palletization process has been largely affected by automation. The load planning is very difficult when it comes to its stability and an efficient usage of the loading area. Nowadays the process of automated palletizing is a very important part of the production system. This article introduces a new software product created especially for the palletizing process planning. This software is called Pallet Sjf STU and is developed at the Faculty of Mechanical Engineering (Slovak University of Technology in Bratislava). The whole software is programmed in Java programming language. Its main aim is to find optimal deployment of objects according to the pallet dimensions. The criterion of optimality is the maximal number of objects stored on one pallet. Here are implemented several tools for the creation of the whole load and the control of its quality. Quality control is based on calculations and algorithms to find collisions between objects. In the next part of

Received February 12, 2012
* **Acknowledgements**. This problem is solved in VEGA MŠ SR 1/0274/11 project.
Corresponding author: Dušan Kravec
Faculty of Mechanical Engineering, Slovak University of Technology in Bratislava, Slovakia
E-mail: dusan.kravec@stuba.sk

this article the modules for deployment of objects, collision control, virtual reality and robot control module are described.

2. Deployment of Objects

The main task of the software is to design an optimal layout of objects on the pallet. Thanks to the implemented methods of deployment the user can choose the best layout. The first step to generate results is to enter this input data:
- Pallet dimensions (load length, width and maximum height)
- Type of object (prism or cylinder)
- Object dimensions (length, width, height of the prism / height and diameter of the cylinder)

The analysis can start after entering all necessary input data. Software evaluates the entered data and chooses the optimal deployment method thanks to implemented algorithms. The optimality criterion is the highest number of objects stored on one pallet. The deployment method that can store up maximum of objects is chosen automatically. If the user is not satisfied with automatically selected result then he can manually select deployment type of method.

For prism – shaped objects the following methods are well known:
- 1 block, 2 block and 3 block method
- Steudls 4 block method, and,
- Smith and DeCannis 4 block method.

For cylinder – shaped objects the following deployment methods are best known:
- "Raster" method, and,
- "Cross" method.

The next step is the calculation of coordinates. Its output are three coordinates (x, y and z) and orientation (horizontal / vertical) of the object. For each object three coordinates and one orientation are generated. In the following sections the well known deployment methods of objects on the pallet are described.

2.1. Single block, 2 block and 3 block method

The method using a single block is the simplest method for resolving the pattern of prisms on the palette. There are only two possible options to place the prisms: *vertical* and *horizontal*. Block dimensions are identical to dimensions of pallet. Calculation of coordinates for horizontally oriented prisms is (2):

$$P_{i,j} = \left[\frac{l}{2} + l(i-1);\ \frac{w}{2} + w(j-1);\ H \right] \tag{1}$$

Calculation of coordinates for horizontally oriented prisms is (2):

$$P_{i,j} = \left[\frac{w}{2} + w(i-1); \frac{l}{2} + l(j-1);\ V \right] \tag{2}$$

Where:

L – the length of the pallet l – the length of the prism
W – the width of the pallet w – the width of the prism
With these conditions: $L \geq W$ and $A \geq B$

The method for using 2 or 3 blocks (Fig. 1) is basically an extension of the single block method. In the method of 2 blocks we are trying to find the best combination of vertically and horizontally oriented prisms along the length of the pallet. Optimum combination contains the smallest unused space between prisms. We can also use the third block to fill in an empty area above the second columns (columns of vertically oriented prisms) using inverted prisms.

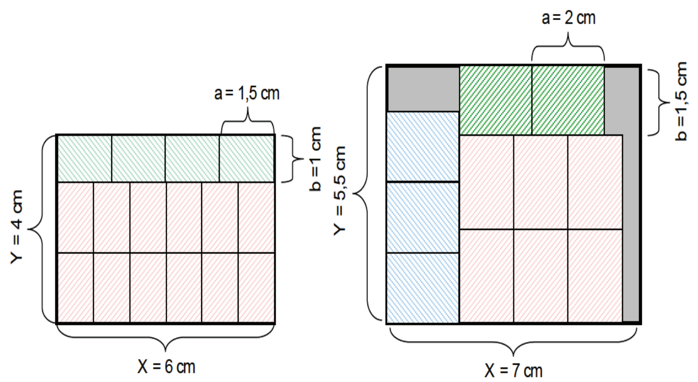

Fig. 1 Two – block and three – block deployment

Analogically we can use this method for the width of the pallet and find the best solution for placing the prisms. The advantage of these methods is the simplicity and clarity. The disadvantage is that it does not provide an optimal solution for every problem.

2.2. Steudls 4 block method

This method divides the storage area for four blocks located in the corners of the pallet. These blocks are rotated, depending on the direction of stored objects (horizontal or vertical). This method was described by Harold J. Steudl in 1979; therefore it is also known as the Steudl algorithm. It is a recursive method using dynamic programming. Dynamic programming is used for process optimization. It divides a big problem into small sub problems. These sub problems are then solved and the results for future potential use are stored.

The first step of this method is finding an efficient combination of horizontally and vertically oriented prisms on a circuit of pallet. In other words, we are trying to find the smallest gap between combination of horizontal and vertical oriented prisms stored in length (width) of pallet. We can do this by using objective function (3):

$$\begin{aligned} &F_n(S_n) = Max[X_n * l + Y_n * w + F_{n-1}(S_{n-1})] \\ &X_n * l + Y_n * w \le D_n \\ &n = 1, 2, 3, 4 \end{aligned} \tag{3}$$

Where:

$F_n\ (S_n)$ – max sum of horizontally and vertically oriented prisms on n side with state variable Sn at the beginning of the side.

X_n – number of objects of length L placed along side n

Y_n – number of objects of width W placed along side n

D_n – length of the pallet

S_n – is a state variable that defines the initial conditions for side n.

This state variable can take these three values:

- Prisms are only horizontally oriented along side n of the pallet
- Prisms are only vertically oriented along side n of the pallet
- Prisms are vertically and horizontally oriented along side n of the pallet.

Objective function is calculating each combination of three values from Sn. For the three possible values of Sn and four blocks of prisms we can use this calculation: $3^4 = 81$. So we have to choose the best result from 81 results. Objective function $Fn\ (Sn)$ can be described as maximizing the utilized length of each side. However, we have to include the pattern of prisms on the previous side. It can also be defined as minimizing the unused circuit of the pallet.

In the second step we have to fill in the unused area. This step is linked by 2 problems. The first one is filling of empty area which can hold one or more prisms. The second problem is overlap of individual blocks which can be identified, for example, in this case (4):

$$D_1 - X_1 l < X_3 l \quad \text{and} \quad D_4 - X_4 l < X_2 l \tag{4}$$

Steudls method is very good for creating an efficient pattern of prisms on pallet, but we have to be careful not to create an overlapped area.

2.3. Smith and Decanis 4 block method

Fig. 2 shows the layout of prisms on the pallet according to Smith and Decannis method. It is investigating all the possible combinations of the shown layout. At first sight, this method is similar to the Steudl one, but the number of objects in the blocks 1-3 and 2-4 is not equal. The principle of determining the number of objects across the width and length of each block is different. The first step is to define the first block. The second block needs to be higher than the first one. The third block needs to be wider than the second one and the fourth block is created in the remaining empty space. All the possible dimensions of the first block as well as those of the other blocks are calculated. Then we can choose the best solution for our dimensions of prisms and pallet. This solution contains the highest number of prisms which we can store on one pallet.

Objective function for this method is (5):

$$MaxZ = a*b + c*d + e*f + g*h \tag{5}$$

Optimization is finished after generating all possible combinations. This method does not allow overlapping of blocks, as it can occur in the Steudl method, but we can find more unused space between blocks. This problem can be solved either by adding another block or several blocks into this empty space.

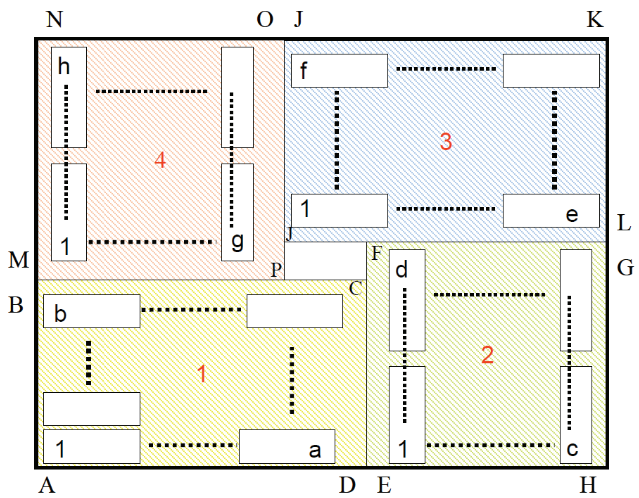

Fig. 2 Smith and DeCannis deployment method

2.4. Cylinders

In terms of methodology, the cylinder is the easiest shape for planning the layout on the pallet. Fig. 3 shows two simple layouts of cylinders, which are not so difficult. In the first case we can see losses in-between the cylinders. In the latter case we are trying to eliminate these losses but not always successfully. This case is effective only if the number of cylinders in the first row equals to that of the cylinders in the second row. We cannot say which of these layouts is more efficient because sometimes the number of cylinders on the pallet is equal in both of them. More advanced software can evaluate both layouts and give the best solution to the user.

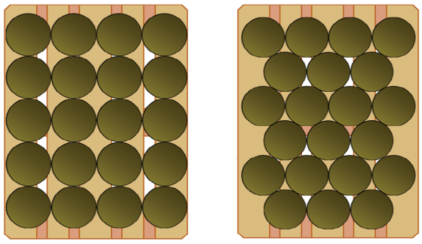

Fig. 3 Deployment patterns of cylinders

Calculation of coordinates for "raster" alignment of cylinders (6):

$$P_{i,j} = \left[\frac{d}{2} + d(i-1); \frac{d}{2} + d(j-1) \right] \tag{6}$$

Calculation of coordinates for "cross" alignment of cylinders (7):

$$P_{i,j} = \begin{bmatrix} d - \frac{d}{2}\left(2*\left[\frac{j}{2}\right] - j\right) + d(i-1); \\ \frac{d}{2} + d*\frac{\sqrt{3}}{2}(j-1) \end{bmatrix} \tag{7}$$

After resolving the question of layout, it is necessary to answer the question of stability. Between the layers of cylinders the pads are inserted that enhance the stability of the whole system. It is inappropriate to combine both types of layout between the layers. To ensure stability, we are using different accessories, such as walls, fences, packing washers, belts, etc.

3. Two – Dimensional Editor

Creating an object layout by implemented algorithms from software is very limiting. Users mostly want to manually change generated positions of the objects. The two–dimensional editor (Fig. 4) gives the opportunity to perform those changes. It is a separate window with workplace, table and control buttons. Generated coordinates of objects are the input data to this window. New coordinates and orientation are generated for each object after editing and this information is the output from window. Coordinates for each object are written in the table. So number of rows in this table equals to number of objects. Columns are coordinates (x, y and z), orientation (0°, 90°) and index of the object. New coordinates and orientations are generated after editing positions.

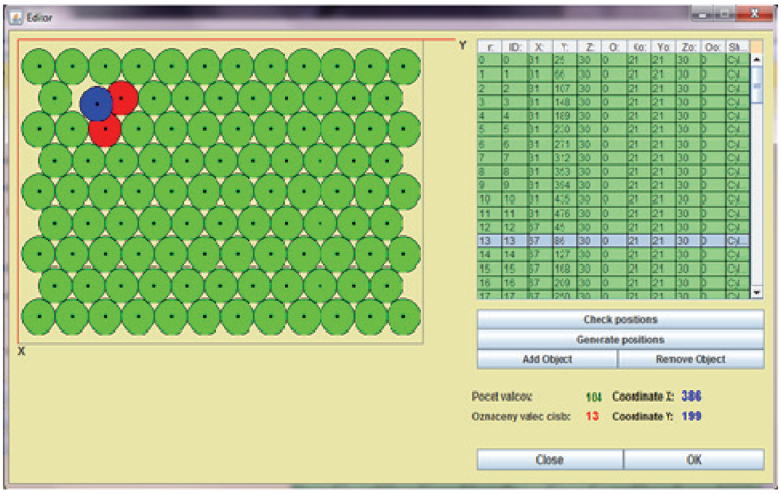

Fig. 4. Two - dimensional editor - Steudl algorithm (four block algorithm)

The workplace is an important part of this window. There is a two-dimensional preview of the objects stored in the layer as well as an outline of the pallet dimensions. We can manipulate objects by using the mouse "drag & drop" method and control keys of keyboard. Objects which require change of orientation (e.g. prism-shaped objects) are rotated by mouse-clicking. Each object has its own JPanel (part of Java JRE library). All JPanels are managed by JLayeredPane() – it is a good control system that allows easy manipulation of objects. Controlling objects by keyboard has more accuracy. SPACE button is used for changing the object selection; therefore, the index of object is very important here. ARROWS are used for manipulating objects. In some cases it is necessary to input a new object of another shape (in mixed loads) to a workspace. That's possible by clicking on ADD OBJECT button and selecting from menu wanted shape.

4. Virtual Reality

Coordinates of the end effector position do not provide an excellent idea of the chosen solutions. Therefore, it is necessary to incorporate the graphic element into the program which would indicate a solution in three – dimensional (3D) preview. Two – dimensional (2D) rendering would be sufficient when we are analyzing only one layer of the load. For multilayer solutions 3D models are better, so it is necessary to use virtual reality interface (Fig. 5). There are many interfaces that can provide rendering of the objects in space. We chose Java3D interface, which includes well known OpenGL GLUT library. This interface is mainly used for modeling components, technical applications and technical product development. It can provide rendering the object in space and it allows its manipulation. It is also used in computer games and simulations. This interface can be controlled by programming languages like C++ and Java. Before starting the program we have to install Java3D library. This library contains forms, shapes, functions, materials, scenes, movements, calculations, etc. The biggest advantage of Java3D interface is that we can have 3D model in the same window with GUI (Graphical User Interface) components like buttons, text boxes, labels, etc. For this purpose we have used two JPanel components. The first contains command buttons with labels and the second is used as canvas for 3D preview.

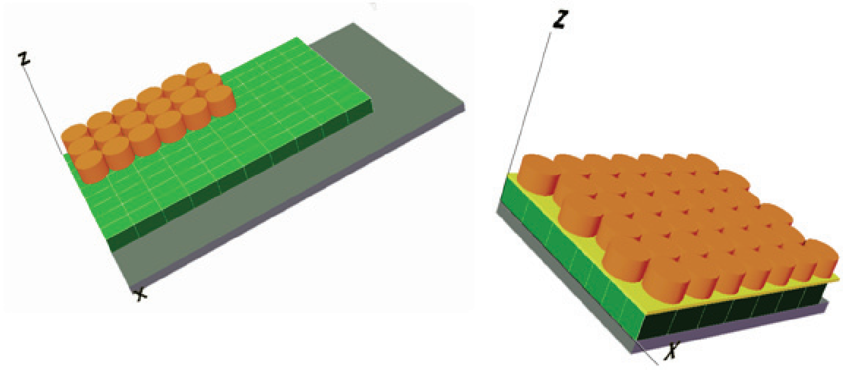

Fig. 5 Virtual reality 3D model

Dimensions of the object, dimensions of the pallet and coordinates with the rotations of objects are used as input data to the model. After drawing the solution into a virtual reality model the user can rotate and control the load by mouse. This module can draw a static or dynamic model of the load. The static model has no movements but we can control it with the mouse. The dynamic model is still rotating.

5. ROBOT CONTROL

An inevitable function of this software is its communication with the SCARA robot. The control system knows several commands for controlling the robot movements, for example: MOVE 100, 100, 90, 50 (go to coordinates 100, 100, 90, 50). Serial link is used for sending the individual commands and also for receiving the answers from the control system. This connection is made by the USB port. We have created a simple window to control the robot from our computer (Fig. 6). It contains a text line to send simple commands, a text field to write the entire program for the robot and a history text field which contains sent commands. Another text field is used for writing the answers from the robot control system. If we will send the entire program from the first text field to the control system, the error will occur because the buffer will be full. The control system buffer can handle only 50 lines of the program at once, so it is necessary to send the entire program line after line to it. To solve this problem we have created two synchronized threads. The first thread sends individual commands to the control system of the robot (output) and the second thread is reading replies from it (input). Thread output is inactive until the thread input does not catch the answer from control system of the robot. After catching the answer, the next line of the program is sent by thread output and again waits for the response.

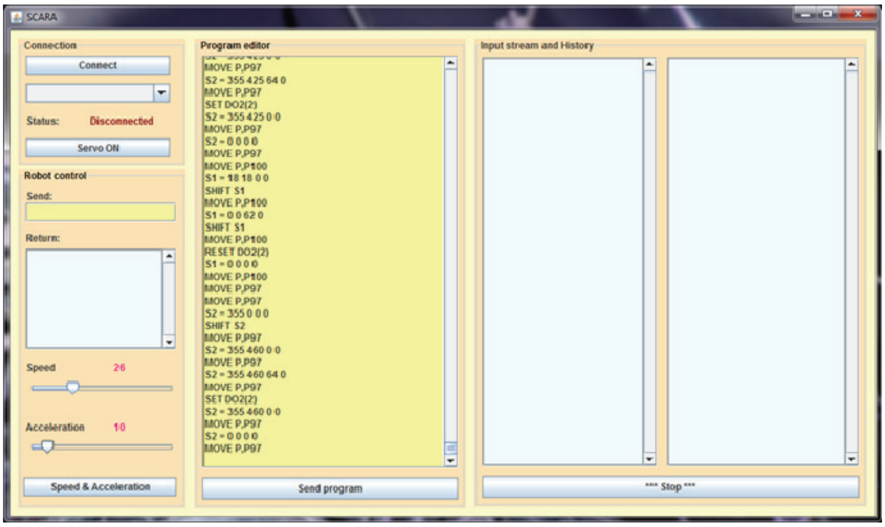

Fig. 6 Robot control module

The whole process will be repeated until the entire program is sent line by line. The second problem is the automatic generation of the commands for robot. For this we have created template commands to store only one basic object. To this template the variables to the coordinates are placed. These variables are automatically changed after each cycle of storage. The control program is reading them from a table that imposes a program for planning the layout of objects. Each coordinate has a characteristic variable. According to the number of lines in this table we determine the number of cycles. If an error occurs in the control program of the robot we can start it again from the last performed command. The last part of the window is a line that informs the connection status to the robot control system via USB. If the connection is ok, then the green sign CONNECTED is shown. If an error occurs then the red sign DISCONNECTED is shown. Refreshing the connection can be done through the CONNECT button.

6. Conclusion

Knowledge of the methods used for two-dimensional layout is essential for creating three-dimensional and mixed loads. Mixed load consists of one or more shapes. Many software products have implemented these methods in their algorithms. They use them for evaluation, calculation and creating optimal layout of objects on the pallet. When three – dimensional distribution is used then it is necessary to create a virtual reality model of pallet load. Thanks to this model we can analyze load and fill the empty area using the methods for two – dimensional loading.

REFERENCES

1. Cormen, T. (2009) Introduction to algorithms, Massachusetts Institute of technology.
2. Hajduk, M. (2008) Robotické bunky, SjF,Košice.
3. Heinze, A. (2006) Optimisation of BMW Group Standardised Load Units via the Pallet Loading Problem, Linkoping.
4. Kuhn, T. (1999) Automatisierte Palettierung mit Mehrfachgreifern, Universitat Hannover, Berlin.
5. Nelissen, J. (1993) New Approaches to the Pallet Loading Problem, Aachen.
6. Yang, M. (1993) Multi-layer palletization of multi-size prismes for 2D and 3D problems, Montreal.

PROGRESIVNE METODE I ALATI KORIŠĆENI U NOVOM PALETSKOM SOFTVERU

Ovaj rad opisuje softver koji se koristi u procesu automatizovanog paletizovanja. Pored primenjenih metoda realizacije, postoje i druge jake alatke kao što su dvodimenzionalni povlačeći editor i trodimenzionalni pregledač i editor (virtualna stvarnost). Editovanje novih položaja objekta suočava se sa problemom sudara. On se rešava primenjenim proračunima i algoritmima. Softverski izlazi sadrže koordinate i orijentaciju za svaki objekat. On se sagledava dublje, iznutra, kao i njegove funkcije; predstavlja se i glavna ideja programa i slike GUI-ija (grafičkog korisničkog interfejsa).

Ključne reči: *paletizovanje, automatizacija, softver, metoda realizacije*