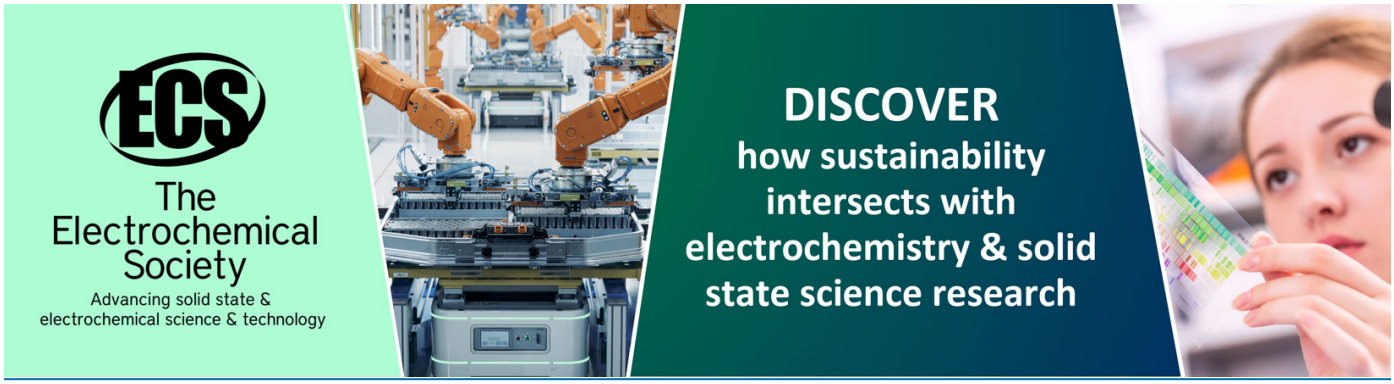
Automatic positioning methodology and algorithm for modular jigs and fixtures components



Automatic positioning methodology and algorithm for modular jigs and fixtures components

T Savu, S Nanu and I C Ene

University POLITEHNICA of Bucharest, Manufacturing Engineering (TCM) Dept., 313 Spl. Independentei, 060042 sector 6, Bucharest, Romania

tom.savu@upb.ro

Abstract. One of the steps in designing a modular jig or fixture assembly is that of selecting the appropriate components. Among these components, those forming the jig's body must be chosen for closing the dimensional chains between the other locating, clamping, or indexing components. The methodology starts by defining the components' virtual connectors, specifying the rules for establishing their coordinates systems, and then is defining the rules to be followed for aligning and assembling two adjacent components. Different cases are described and only some of them are detailed, for clarity reasons. The algorithm is looking to determine a series of components from which the first is in a fixed position, related to other locating or clamping jig's component. A hierarchical tree is generated, using available components, having in each node a component which is assembled with a previous one using the above-mentioned connectors. Interference criteria is checked for selecting possible solutions only. A stop criterion related to the number of components in a branch is used, Solutions from the different branches are compared using the distances to an end component.

1. Introduction

When designing a modular jig or fixture assembly, having a set of available modular components, one has to solve the problem of identifying an optimal subset of modular components which must satisfy, between others, some geometrical conditions, established in most cases for closing dimensional chains between other jig or fixture components.

The paper is describing an approach for T-slot-based systems [1], in which the different possible combinations of components are built by using components connectors, defined in the specific points where which component can be assembled with another component. These connectors are entities like the ones used in different CAD software packages for describing the assembly of two parts.

To apply and implement an algorithm for the automated selection of the optimal subset, data representation structures have to be defined for describing the content of the set of available modular components, the components' dimensions, the position and the orientation of the components' connectors and also for representing the different possible solutions.

The algorithm has to be able to identify possible new solutions, to check for their compatibility with various criteria, to add the compatible solutions to a set, to check for an end of searching condition and then to apply some criteria for selecting the optimal solution.

An implementation was developed using LabVIEW [2], which will later allow storing the data in databases, presenting the solutions in 3D formats, and accessing the application as a Web service.

2. Notation conventions for describing the components

It is considered that a component Ci has two types of virtual connectors (figure 1):

- a main connector, named Ci_ConP, placed on the component's face on which a T-slot nut is mounted on the component:
  - the Z axis of the main connector Ci_ConP is parallel with the axis of the screw which is assembled with the T-slot nut, with the positive sense toward the nut;
  - the X-axis of the main connector Ci_ConP is parallel with the direction on which the T-slot nut can slide;
- none or more secondary connectors, placed on the faces where T-slots are present:
  - the Z-axis of a secondary connector Ci_ConSj is parallel with the axis of the screw which is assembled with the T-slot nut, with the positive sense toward the neighbouring component (outside the parent component);
  - the X-axis of a secondary connector Ci_ConSj is parallel with the direction on which the T-slot nut can slide

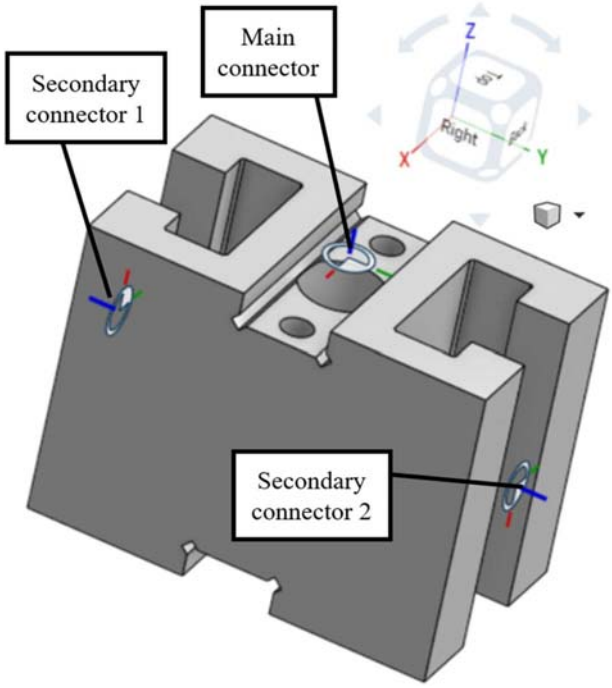


Figure 1. Main and secondary connectors for a CDC 048 type component.

The X(Ci_ConP), Y(Ci_ConP) and Z(Ci_ConP) coordinates of a main connector are expressed using the equipment's coordinate system, while the coordinates of a secondary connector Ci_ConSj are expressed using the main connector's coordinate system.

The rotation angles alfa(Ci_ConSj), beta(Ci_ConSj) and gamma(Ci_ConSj) of a coordinate system belonging to a secondary connector, relative to the main connector's coordinate system, are expressed considering the situation in which the rotations are performed in the X, Y, Z order. The positive sense of a rotation is the counterclockwise one.

One component's dimensions are described using its faces coordinates, expressed in the coordinates system of the component's main connector. Using as an example the CDC 048 type component from the DISROM set [3], its dimensions are Xmin = -19, Xmax = 19, Ymin = -32, Ymax = 32, Zmin = -48, Zmax = 0 (figure 2).

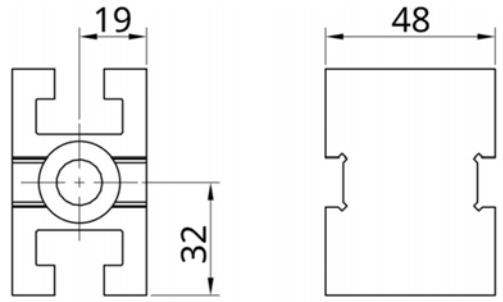


Figure 2. CDC 048 type component dimensions.

Also, the coordinates of the first secondary connector are X(Ci_ConS1) = 0, Y(Ci_ConS1) = -32, Z(Ci_ConS1) = -24, while the coordinates of the second secondary connector are X(Ci_ConS2) = 0, Y(Ci_ConS2) = 32, Z(Ci_ConS2) = -24. The rotation angles of the first secondary connector are alfa(Ci_ConS1) = 90°, beta(Ci_ConS1) = -90°, gamma(Ci_ConS1) = 0°, while the rotation angles of the second secondary connector are alfa(Ci_ConS2) = -90°, beta(Ci_ConS2) = 90°, gamma(Ci_ConS2) = 0°.

3. Assembly rules

When two components C1 and C2 must be assembled, one main connector of one component must be aligned with one secondary connector of the other component. There are two possible cases for aligning the connectors:

- the main connector C1_ConP of the first component is aligned with a secondary connector C2_ConSj of the second component;
- the main connector C2_ConP of the second component is aligned with a secondary connector C1_ConSj of the first component.

For the first case, the following conditions must be met for aligning the two connectors, all expressed in the coordinate system of the C1_ConP connector:

- Z(C2_ConSj) = 0, the origins of the two connectors have to stay in the same X-Y plane;
- (alfa(C2_ConSj) = π and beta(C2_ConSj) = 0) or (alfa(C2_ConSj) = 0 and beta(C2_ConSj) = π), the Z-axis of the two connectors can be made parallel and in opposite directions either using a rotation around the X-axis or using a rotation around the Y-axis;
- Y(C2_ConSj) = 0, no translation on the Y-axis;
- gamma(C2_ConSj) = 0, no rotation around the Z-axis;
- X(C2_ConSj) ∈ [δ1, δ2], limited translation on the X-axis, where δ1 and δ2 depend on the dimensions of the second component.

Further discussing only for the simplified case when X(C2_ConSj) = 0, there are two possible cases for the C2_ConSj relative coordinates in the C1_ConP system:

- X(C2_ConSj) = Y(C2_ConSj) = Z(C2_ConSj) = 0, alfa(C2_ConSj) = π, beta(C2_ConSj) = 0, gamma(C2_ConSj) = 0;
- X(C2_ConSj) = Y(C2_ConSj) = Z(C2_ConSj) = 0, alfa(C2_ConSj) = 0, beta(C2_ConSj) = π, gamma(C2_ConSj) = 0.

Discussing about the first of the two cases above, when the C1_ConP absolute coordinates are known, the absolute coordinates of C2_ConSj can be computed by rotating with π around the X-axis the absolute coordinates of C1_ConP [4].

Similarly, for the second case, the following conditions must be met for aligning the two connectors, all expressed in the coordinate system of the C1_ConSj connector:

- Z(C2_ConP) = 0, the origins of the two connectors have to stay in the same X-Y plane;
- (alfa(C2_ConP) = π and beta(C2_ConP) = 0) or (alfa(C2_ConP) = 0 and beta(C2_ConP) = π), the Z-axis of the two connectors can be made parallel and in opposite directions either using a rotation around the X-axis or using a rotation around the Y-axis;
- Y(C2_ConP) = 0, no translation on the Y-axis;
- gamma(C2_ConP) = 0, no rotation around the Z-axis;
- X(C2_ConP) ∈ [δ1, δ2], limited translation on the X-axis.

Further discussing only for the simplified case when X(C2_ConP) = 0, there are two possible cases for the C2_ConP relative coordinates in the C1_ConSj system:

- X(C2_ConP) = Y(C2_ConP) = Z(C2_ConP) = 0, alfa(C2_ConP) = π, beta(C2_ConP) = 0, gamma(C2_ConP) = 0;
- X(C2_ConP) = Y(C2_ConP) = Z(C2_ConP) = 0, alfa(C2_ConP) = 0, beta(C2_ConP) = π, gamma(C2_ConP) = 0.

Discussing about the first of the two cases above, when the C1_ConSj absolute coordinates are known, the absolute coordinates of C2_ConP can be computed by rotating with π around the X-axis the absolute coordinates of C1_ConSj.

4. Algorithm description

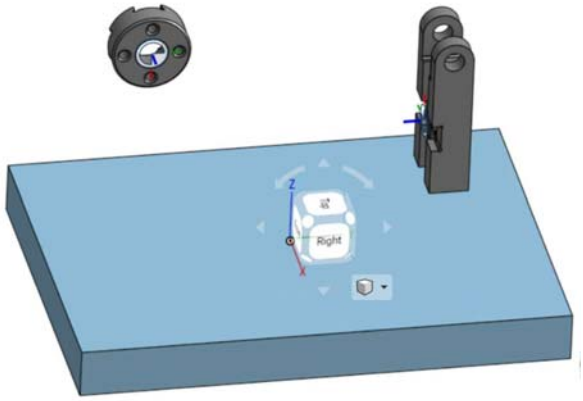


Figure 3. Initial components of the dimensional chain.

The algorithm is starting from the situation when the dimensional chain must be closed between two previously established components (figure 3) and is looking to build, for each initial component, a tree of modular components Ci.

It was assumed, for simplifying the initial cases for which the algorithm was implemented, that these initial components have only one free main connector each.

Each branch of one tree will contain a possible subset of modular components. The tree is initialized with one of the components between which the dimensional chain must be closed.

At a certain moment, there are a number of free connectors in a tree, belonging to the modular components in the subset (figure 4). For each free connector in the tree, the algorithm is looking in a list of available modular components ACk and is trying to extend the tree with more branches, by assembling available modular components on the free connectors.

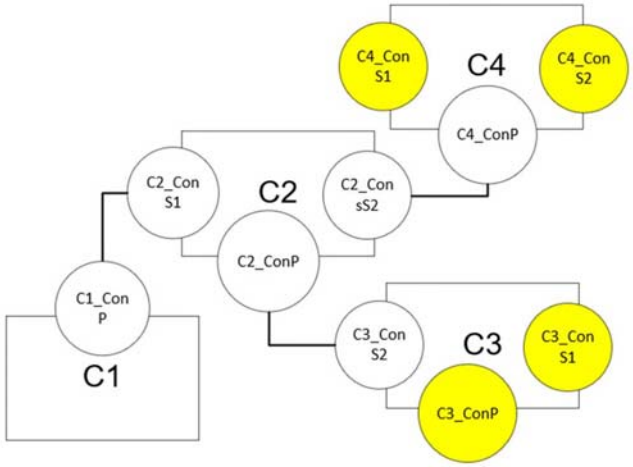


Figure 4. Tree structure, with free connectors marked in yellow.

For evaluating the possibility of assembling an available modular component on a free connector, interference conditions are checked between the available component and all the other components already on the tree branch.

The algorithm is not building branches with more than a specified number of components maxC, this being a limitation imposed by the rigidity of the jig or fixture assembly.

Because the two subsets of modular components are presumed to be assembled on the same base plate, the algorithm is computing the vertical distances between free connectors, belonging to terminal components of branches in the two separate trees, with vertical Z-axis. A stopping condition appears if the distance between a pair of such free connectors is smaller than a predefined value.

5. Data structures definitions

Data structures were defined for describing some of the conventions mentioned before. The data structure used for organizing a connector's linear and angular coordinates is presented in figure 5, while the data structure for describing a component's dimensions is presented in figure 6.

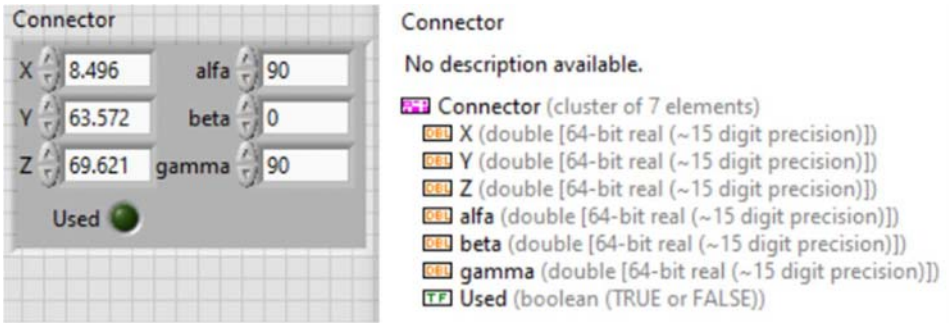


Figure 5. Data structure for connector's linear and angular coordinates.

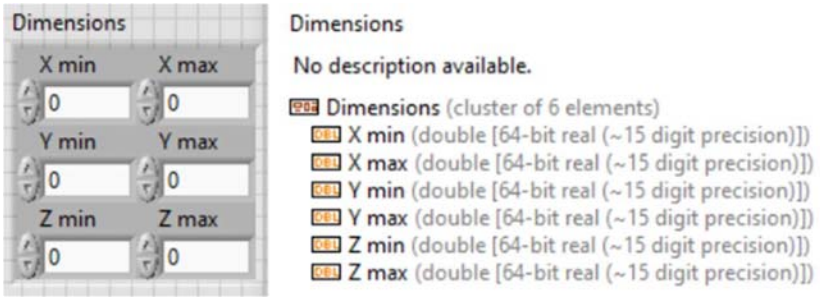


Figure 6. Data structure for describing a component's dimensions.

The data structure for completely describing a component contains substructures for: the component's main connector's coordinates, the component's dimensions, an array containing substructures for the secondary connectors' coordinates and a value specifying the component's type (figure 7).

Figure 7. Data structure for complete component description.

When a component is described as part of the tree which is built by the algorithm, its data structure (figure 8) contains its type, the main connector's coordinates, a substructure containing the connection information, a rank value and an array describing the states of the secondary connectors (used or not). The rank value was not used for the moment but was placed for future developments.

The substructure containing the connection information has three unsigned byte values. If the substructure belongs to a component which has the index j in the tree branch and is connected to a component with index i, the three values are: the index i, the index of the used connector on the component with index i and the index of the used connector on the component with index j.

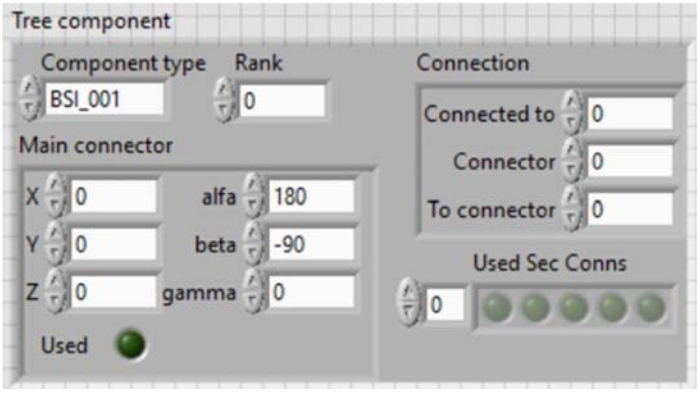


Figure 8. Data structure for a component part of a tree.

When a component is described as part of the set of available modular components, its data structure (figure 9) contains its type, the main connector's coordinates, an array with its secondary connectors' coordinates and an array of availability data.

Because each branch of the solutions tree built by the algorithm represents a possible solution, separate from the other branches, the number of instances of a component of a certain type in the whole tree may be greater than the number of real instances in the set of available modular components. Therefore the availability of a component of a certain type is described separately for each tree branch and the values are stored in an array structure.

The tree branches are stored in a two-dimensional array of components, where each line is representing a branch.

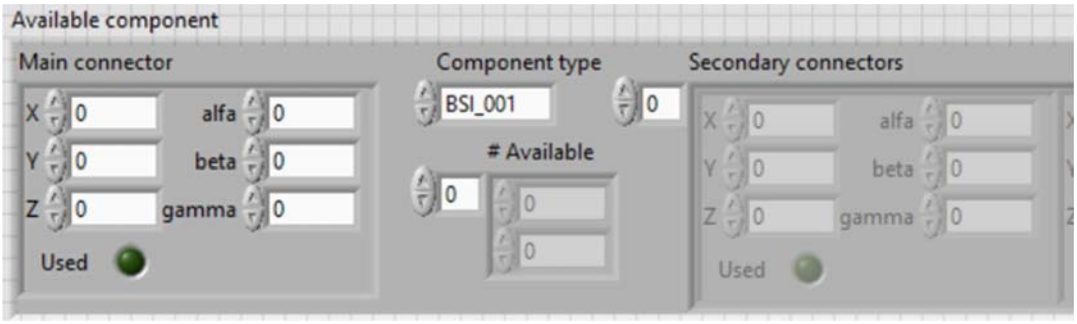


Figure 9. Data structure for an available modular component.

For checking the interference conditions [5] between an available component ACk and another component Cj already on the tree branch where ACk could be placed, with the modular components presumed to be shaped as simple boxes, for each face of ACk and for each edge of Cj, a subroutine is computing the coordinates of the intersection point Pjk between ACk face and the Cj edge. If Pjk belongs to the Cj edge, then it is checked if Pjk belongs also to the ACk face. If it exists at least one combination between an ACk face and a Cj edge for which the above conditions are met, it means that interference exists between the two components, so the tested solution is not acceptable and the ACk component is not placed in the tree.

6. Conclusions

The algorithm was tested for a simple implementation, which is taking into account only some of the assembly cases mentioned in the paper, with only a limited number of types of available components, but the implementation proved to offer correct solutions and is able to be further developed with minimum efforts. Further validation methods which will use 3d representations of the designed solutions will be also developed.

References

[1] Ghatpande P 2008 *Study of Fixturing Accessibilities in Computer-Aided Fixture Design – Master of Science Thesis* (Massachusetts: Worcester Polytechnic Institute)
Rong Y, Huang S and Hou Z 2005 Advanced Computer-Aided Fixture Design (Burlington: Elsevier Inc.)
Rong Y and Zhu Y 1999 *Computer-Aided Fixture Design* (New York: Marcel Dekker Inc.)
Bgoi Kok Ann B 1990 Computer Aided Design of Modular Fixture Assembly – Ph.D. Thesis (Cristchurch: University of Canterbury)
[2] Bitter R, Mohiuddin T and Nawrocki M 2017 *LabVIEW: Advanced Programming Techniques, Second Edition* (Boca Raton: CRC Press – Taylor & Francis Group)
[3] Bragaru A, Panus V, Dulgheru L and Armeanu A 1982 SEFA-DISROM Sistem si metoda, Volumul I - *Teoria si practica proiectarii dispozitivelor pentru prelucrari pe masini-unelte* (Bucharest: Editura Tehnica)
[4] Bona B 2015 *Reference Frames and Rotations* (Turin: Politecnico di Torino)
[5] Hughes J, van Dam A, McGuire M, Sklar D, Foley J, Feiner S and Akeley K 2013 *Computer Graphics: Principles and Practice* (USA: Addison-Wesley Professional)

Acknowledgments

The authors were inspired to start working on this subject during the support they offered to Mr. Marian Claudiu Ursei during the development of his graduation project. Mr. Ursei, who also developed and presented an own method, showed interest, and appreciated that the subject will be promising.