Journal of Advanced Manufacturing Systems
Vol. 10, No. 2 (2011) 277–291

DOI: 10.1142/S021968671100220X

AUTOMATIC COLLISION DETECTION FOR ASSEMBLY SEQUENCE PLANNING USING A THREE-DIMENSIONAL SOLID MODEL

ALFADHLANI

Industrial Engineering Department
Andalas University, Limau Manis
Padang, 25163, Indonesia
alfadhlani@ft.unand.ac.id

T. M. A. ARI SAMADHI*, ANAS MA'RUF† and ISA SETIASYAH TOHA‡

Industrial Engineering Department
Bandung Institute of Technology, Jl. Ganesha 10
Bandung, 40132, Indonesia
**samadhi@mail.ti.itb.ac.id*
†maruf@lspitb.org
‡isa_st@polman-bandung.ac.id

Assembly sequence planning of a product involves several steps, including generation of precedence constraints, generation of assembly sequences, and selection of assembly sequences. Generation and selection of assembly sequences should be able to guarantee the feasibility of assembly. Assembly will be feasible if there is no collision between components when assembled. Detection of collision-free path of assembly can be done in an automated way. There are a number of collision detection methods that have been developed, but the method requires a complicated process of data geometry analysis. This paper proposes a method for detecting a collision-free path of the assembly component in a more simple way. Geometrical data required, taken from the three-dimensional (3D) solid drawing in the form of stacked drawing in computer-aided design (CAD) systems. Retrieval of geometrical data of components and detection of the collision-free path of an assembly were done in an automated way, directly from the CAD system.

Keywords: Collision detection; Assembly sequence; CAD.

1. Introduction

More than 70% of product manufacturing cost is determined during the design stage.[1] Using CAD system in product design provides the possibility of product evaluation at the early design stage. Some possibilities are evaluation of component, product, tool, and fixture design.[2] Assembly is a major problem in the manufacturing process because it affects the design, engineering, manufacturing, and logistics.[3]

There are a number of feasible sequence alternatives to assemble a product. The selection of such alternatives is done by a manufacturing engineer considering some criteria such as practicality or other limitations.[4] Assembly sequence is feasible if there is no collision between components. Precedence constraint is a tool to generate feasible assembly sequence of a complex product.[5] The assembly planner must consider the collision problem to generate the constraints, which contains the information about the predecessor and successor of the components to be assembled. Delchambre[6] distinguished two types of precedence constraint as follows:

(i) Hard constraints
Hard constraints must be considered in generating assembly sequences because of the requirement of component geometry and component position on the final assembly. There are three classes of hard constraints, namely geometrical constraint, mechanical constraint, and component constraint. Geometrical constraint is related to the interference problem between components. Mechanical constraint is related to the use of fasteners that must be accessible and not obstructed by other components. Component constraints are used to ensure the tool used in assembly; for example, in bearing assembly process, special tools are required.

(ii) Soft constraints
If the resulting sequence is feasible without considering soft constraints, then the constraints can be eliminated. There are two types of soft constraints, i.e. stacking constraints and technological constraints. Stacking constraints are caused by external fasteners. For example, a number of components that are fastened together with the same screw should be assembled in a stack position. Technological constraints are determined by the designer, which arise from the use of tools to make a product.

In 1984,[6] Bourjault proposed a number of interactive procedures by compiling a series of questions to generate precedence constraints. The analysis is based on a number of l-connection (*liaison*) between components. There are three stages in the analysis of connections, and each stage consists of two questions. If there are five connections between components, then the number of questions to be asked is about 60–160 in order to generate the assembly precedence relations. De Fazio and Whitney[7] conducted research to simplify the procedure proposed by Bourjault and reduced the number of questions to $2l$.

Precedence constraints that refer to the problem of geometry can be formed by detecting a collision-free path of assembly. Delchambre[6] detected a collision-free path of assembly with three steps: examining the intersection of projections *parallelipipedic* envelopes, examining the intersection of apparent outlines, and searching for obstacle facets. Gu and Yan[8] detected a collision-free path of assembly based on information of contact between components represented in the form of graph. Smith, Smith, and Liao[9] proposed a method for detecting a collision-free path of assembly based on the information of interference-freeness matrix. Smith method was used by Pan, Smith, and Smith.[10] in developing automated methods of assembly sequence planning from STEP CAD files, where interference-freeness analysis is done by the projection geometry data components (edge, vertex, face) from 3D to 2D along the six-axis. By using this approach, the collision between components is found if the projection of two components overlaps in one direction.

The collision detection methods mentioned earlier require a complex process in analyzing geometry data since the CAD system cannot define the mating conditions between components automatically. This paper proposed an automated method to detect collision-free paths of the assembly component in the CAD system. Using the method, the CAD system will be able to define the mating conditions automatically. The CAD system used in this paper is *SolidWorks* 2005 and stacked drawing in a 3D solid model is used as an input.

The rest of the paper is composed as follows: Sec. 2 discusses the relational information, Sec. 3 explains the model development, Sec. 4 provides an algorithm implementation and the conclusion is given in Sec. 5.

2. Relational Information

Assembly product consists of a number of components that support the product functionality. These components are in contact with other components. These contacts are defined as relational in this paper, which are expressed as a mating.

Mating between components faces consist of several types: *against*, *fit*, *contact*, *tight fit*, and *coplanar*. *Against* type is a condition in which two planar faces are in touch and their normal directions are opposite to each other. *Fit* type is defined for two cylinder faces in contact. *Contact* type is a condition that prevents *against* type to move in one or more directions particular. *Tight fit* type is a condition that prevents *fit* type from moving in one direction or more (usually rotational movement), and *coplanar* type is a condition in which two planar faces are on the same plane (the normal direction of the two faces is the same).[11]

SolidWorks 2005 has seven standard mating types, which are *coincident*, *parallel*, *perpendicular*, *tangent*, *concentric*, *distance*, and *angle*.[12] *Coincident* mating type is similar to *against* mating or *coplanar* mating type, depending on the normal direction of contacted faces. *Parallel* mating type is a condition in which two contacted faces are parallel to each other. *Perpendicular* mating type is the relationship of two faces that are mutually perpendicular. *Tangent* mating type is a condition in which two contacted faces intersect at one point. One of the contacted faces should have a cylindrical, conical or spherical shape. *Concentric* mating type is based on the similarity of axes line between components, similar to the *fit* mating type. *Distance* mating type is similar to the *coincident* mating type, except the *distance* mating type has a certain distance between contacted face. *Angle* mating type has a certain angle to the face of contacts.

The existence of mating on components will reduce the number of degrees of freedom (DOFs) of the component. The number of DOFs depends on the type of mating and the number of components in contact. Thus, the mating can be used to determine a collision-free path for each component.

There are six reductions of DOFs on components with coincident mating type. The component can be moved translational along the all axes, except the axis of normal vector of contacted components. Suppose that the normal vector is on the y-axis, so translational movements can be done along the x- and z-axis, while the rotational movement can be done only on the axis where the normal vector is located. Thus, the component will have the number of DOF as many as six directions as shown in Fig. 1.

Designers must consider the suitability of mating placement on the designing component in *SolidWorks*. Mating can be done on a component face only if there is physical contact between components, or if the component position on the final assembly causes the blocking of other component movements. Sometimes, designers use a mating to facilitate the process of assembly in *SolidWorks* in a wrong way. Therefore, we need editing after assembly drawing is completed. The use of mating will influence the results of collision detection analysis.

There is no difference in the drawing appearance caused by the difference in way of mating affixing to the components. As shown in Fig. 2(a), there are three

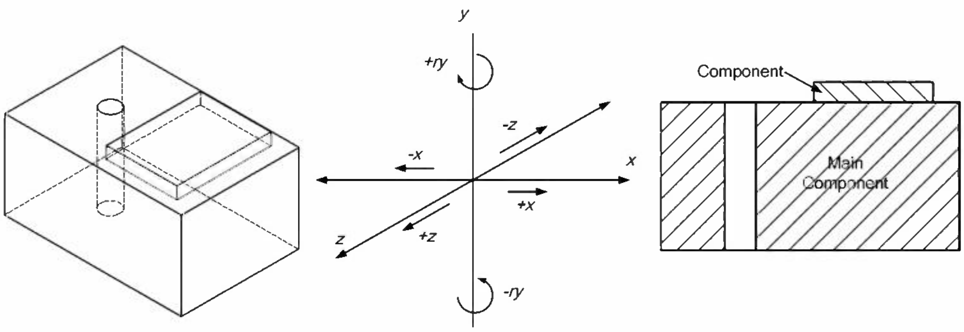


Fig. 1. The DOFs on components with coincident mating type.

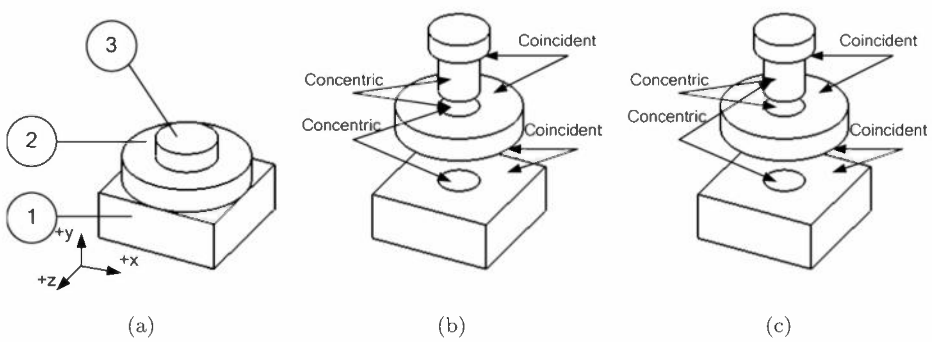


Fig. 2. Affixing a mating on solid model.

components, namely 1, 2, and 3. Figure 2(b) and 2(c) show the affixing of mating to the face in a different way. In Fig. 2(b), concentric mating type are given to the cylinder face of component 3 in a contact to a hole face of component 1, and contact between a hole face of component 2 with a hole face of component 1. Affixing a mating this way causes a wrong interpretation that there is no contact between the cylinder face of component 3 with a hole face of component 1 that should be exist, and there is a contact between a hole face of component 1 with a hole face of component 2 that should not be exist. The correct way to affixing a mating should be as in Fig. 2(c). Although the way to affixing a mating in Fig. 2(b) and 2(c) are different, both appear the same as Fig. 2(a).

3. The Development of Collision Detection Model

Collision detection analysis method developed in this paper, refers to the method proposed by Smith, Smith, and Liao.[9] If the translational movement of component i along x-axis, blocked by component j, then the value of the collision-free path to assemble component i on $+x$ equals to 0 (see Fig. 3). Thus we conclude that if a translational movement of a component along k-axis blocked by other components, then the value of a collision-free path to assemble that component on k-axis is 0 and 1 otherwise.

Collision-free path to assemble components in Fig. 2(a) can be determined as shown in Table 1. The AND logic is used to evaluate the final collision-free path of each component. If components 2 and 3 have been assembled, then component 1 can be assembled in $+y$ direction. If components 1 and 3 have been assembled, then

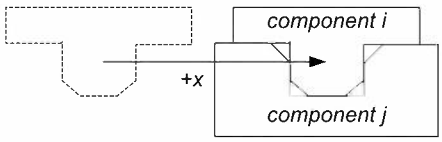


Fig. 3. Collision between components.

Table 1. Collision-free path to assemble component.

Component	Contact	$+x$	$-x$	$+y$	$-y$	$+z$	$-z$
	1,2	1	1	1	0	1	1
1	1,3	0	0	1	0	0	0
	AND	0	0	1	0	0	0
	2,1	1	1	0	1	1	1
2	2,3	0	0	1	0	0	0
	AND	0	0	0	0	0	0
	3,1	0	0	0	1	0	0
3	3,2	0	0	0	1	0	0
	AND	0	0	0	1	0	0

component 2 cannot be assembled in any direction. Component 2 can be assembled with component 3 in $+y$ direction.

This paper defines the collision-free path of assembly based on mating type of component face that contact along the six-axis coordinate which considered: $+x, -x, +y, -y, +z$, and $-z$. Mating type and contacted face information of components are obtained from CAD database using the Algorithm of Database Component Formation as described in the appendix. Efforts to evolve full automation can be done because the information needed can be obtained automatically. The collision-free path of assembly is defined based on mating type and face shape of contacted component as described in the following propositions:

Proposition 1. *For a pair of components that come in contact on planar surfaces using the type of mating: coincident, parallel, or distance, and for a pair of component that contacts on a cylindrical or a planar surfaces using tangent mating type, their collision-free path of assembly can be determined as follows*:

(a) *If the normal vector* N *is* $k = +x, +y, +z$, *then the assembly from* $-k$ *direction is* 0 *and for others it is* 1.
(b) *If the normal vector* N *is* $k = -x, -y, -z$, *then the assembly from* $+k$ *direction is* 0 *and for others it is* 1. □

Proposition 2. *For a pair of components that come in contact on a conical surface using coincident mating type, their collision-free path of assembly can be determined as follows*:

(a) *If the normal vector* N *is* $k = +x, +y, +z$, *then the assembly from* $+k$ *direction is* 1 *and for others it is* 0.
(b) *If the normal vector* N *is* $k = -x, -y, -z$, *then the assembly from* $-k$ *direction is* 1 *and for others it is* 0. □

Proposition 3. *For a pair of components that come in contact on a cylindrical surface with a cylindrical surface or on a cylindrical surface with a conical surface, using concentric mating type, their collision-free path of assembly can be determined as follows*:

(a) *If the normal vector* N *is* $k = +x, +y, +z$, *then the value of* $+k$ *and* $-k$ *direction is* 1 *and for others it is* 0.
(b) *If the normal vector* N *is* $k = -x, -y, -z$, *then the value of* $+k$ *and* $-k$ *direction is* 1, *while for others it is* 0. □

Based on the propositions, we define a set of rules for determining the collision-free path of assembly for each component. The rule is then used in the Algorithm of Collision Detection Analysis.

The rules of collision-free path of assembly:

(1) If a component is assembled to its end position, and has contact with other components, the contact occurs on a planar surface with a planar surface using

the type of mating: *coincident*, *parallel*, *distance*, or on a planar surface with a cylindrical surface using tangent mating type, then check the direction of normal vector of that component.

(a) If the direction of normal vector is $+k$, then the assembly from $-k$ direction is 0 and for others it is 1.
(b) If the direction of normal vector is $-k$, then the assembly from $+k$ direction is 0 and for others it is 1.

(2) If a component is assembled to its end position, and has contact with other components, the contact occurs on a conical surface with a conical surface using coincident mating type, then check the normal vector direction of that component.

(a) If the direction of normal vector is $+k$, then the assembly from $+k$ direction is 1 and for others it is 0.
(b) If the direction of normal vector is $-k$, then the assembly from $-k$ direction is 1 and for others it is 0.

(3) If a component is assembled to its end position, and has contact with other components, the contact occurs on a cylindrical surface with a cylindrical surface, or on a cylindrical surface with a conical surface, using concentric mating type, the direction of normal vector of that component is then checked.

(a) If the direction of normal vector is $+k$, then the assembly from $+k$ and $-k$ directions is 1, while for others it is 0.
(b) If the direction of normal vector is $-k$, then the assembly from $+k$ and $-k$ directions is 1, while for others it is 0.

The collision detection algorithm is developed based on the logic of component detection collision, which has been described in Propositions 1 to 3. The algorithm is started with the component that has the highest number of connections. This is merely to maintain order so that the analysis is done in a random way and this is not a priority rule that affects the outcome. The algorithm collision detection is as follows:

The algorithm of collision detection:

(1) Get all component names from the component database and then specify the number of connections of each component. Save in a component list.
(2) Get one component that has not been analyzed and the list of its mating. Begin analysis with a component that has the highest number of connections.
(3) Get one mating to be analyzed from the mating list of component being analyzed, check i, the normal vector direction of components, on this mating with a value $+1$ or -1.
(4) Check the type of that mating, if the mating type is parallel, coincident, tangent, distance, or concentric, then proceed to the next step, otherwise stop iteration.

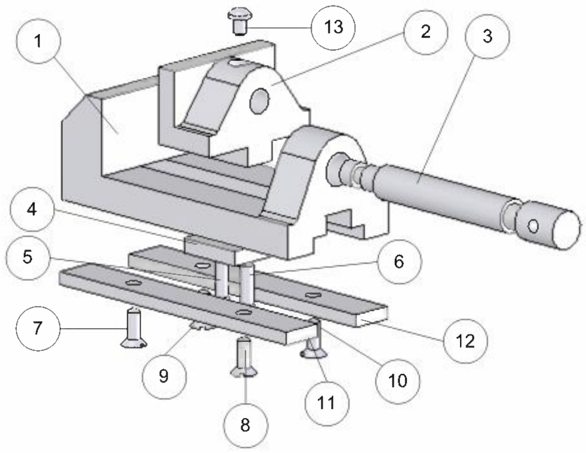
(5) Determine the value of collision-free path using the rule of collision-free path of assembly and then save the results.
(6) Check the list of mating components being analyzed, if there is still a mating component that has not been analyzed then go back to step 3, otherwise continue to the next step.
(7) Evaluate the value of collision-free path of all components with AND logic in six directions considered, save the results on the Collision-Free Path Database of Components.
(8) Check if there are components that have not been analyzed then return to step 2, otherwise the iteration is finished.

The result of this collision detection algorithm is the collision-free path of each component assembly. These results are stored in Collision-Free Path Database of Components.

4. Algorithm Implementation

Bench Vice assembly is used to test the algorithm. The assembly orientation is multidirectional and orthogonal to the x, y, and z axes. Figure 4 shows the Bench Vice which is adopted from Tickoo,[12] and redrawn for the testing purposes. The Bench Vice consists of 13 components with 38 mating. Component list, mating type, and the direction of normal vector data were obtained from the CAD database (*SolidWorks*) using the algorithm of database formation of component geometry. The data are shown in Table 2.

Using the data in Table 2 as input and assuming that all Bench Vice components have been assembled, then based on mating information, the collision-free path of assembly for each component in six directions being considered can be determined.



No	Component
1	Vice body
2	Vice jaw
3	Jaw screw
4	Clamping
5	Set screw 2.1
6	Set screw 2.2
7	Set screw 1.1
8	Set screw 1.3
9	Set screw 1.2
10	Set screw 1.4
11	Base plate 1
12	Base plate 2
13	Oval fillister

Fig. 4. Bench Vice Assembly.[12]

Table 2. Mating and normal vectors information of Bench Vice.

No	Component		i	j	k	Mating type	No	Component		i	j	k	Mating type
1	1	Vice Body	0	1	0	*Coincident1*	20	11	Base plate-1	0	1	0	*Concentric12*
	2	Vice Jaw	0	−1	0	*Coincident1*		7	Set screw 1-1	0	−1	0	*Concentric12*
2	1	Vice Body	0	1	0	*Coincident2*	21	11	Base plate-1	0	1	0	*Concentric13*
	2	Vice Jaw	0	−1	0	*Coincident2*		8	Set screw 1-2	0	−1	0	*Concentric13*
3	1	Vice Body	0	0	1	*Parallel1*	22	1	Vice Body	0	−1	0	*Concentric14*
	2	Vice Jaw	0	0	−1	*Parallel1*		7	Set screw 1-1	0	−1	0	*Concentric14*
4	1	Vice Body	0	0	−1	*Parallel2*	23	1	Vice Body	0	−1	0	*Concentric15*
	2	Vice Jaw	0	0	1	*Parallel2*		8	Set screw 1-2	0	−1	0	*Concentric15*
5	1	Vice Body	−1	0	0	*Parallel3*	24	1	Vice Body	0	−1	0	*Coincident13*
	2	Vice Jaw	1	0	0	*Parallel3*		11	Base plate-1	0	1	0	*Coincident13*
6	1	Vice Body	1	0	0	*Parallel4*	25	11	Base plate-1	0	−1	0	*Coincident14*
	2	Vice Jaw	−1	0	0	*Parallel4*		7	Set screw 1-1	0	1	0	*Coincident14*
7	1	Vice Body	1	0	0	*Concentric1*	26	11	Base plate-1	0	−1	0	*Coincident15*
	3	Jaw screw	−1	0	0	*Concentric1*		8	Set screw 1-2	0	1	0	*Coincident15*
8	2	Vice Jaw	1	0	0	*Concentric2*	27	12	Base plate-2	0	1	0	*Concentric18*
	3	Jaw screw	−1	0	0	*Concentric2*		9	Set screw 1-3	0	−1	0	*Concentric18*
9	2	Vice Jaw	0	1	0	*Concentric3*	28	12	Base plate-2	0	1	0	*Concentric19*
	13	Oval Fillister	0	1	0	*Concentric3*		10	Set screw 1-4	0	−1	0	*Concentric19*
10	2	Vice Jaw	1	0	0	*Parallel5*	29	1	Vice Body	0	−1	0	*Concentric20*
	3	Jaw screw	−1	0	0	*Parallel5*		9	Set screw 1-3	0	−1	0	*Concentric20*

Table 2. (*Continued*)

No		Component	i	j	k	Mating type	No		Component	i	j	k	Mating type
11	3	Jaw screw	1	0	0	*Tangent6*	30	1	Vice Body	0	−1	0	*Concentric21*
	13	Oval Fillister	−1	0	0	*Tangent6*		10	Set screw 1-4	0	−1	0	*Concentric21*
12	2	Vice Jaw	0	1	0	*Coincident9*	31	1	Vice Body	0	−1	0	*Coincident16*
	13	Oval Fillister	0	−1	0	*Coincident9*		12	Base plate-2	0	1	0	*Coincident16*
13	2	Vice Jaw	0	−1	0	*Coincident10*	32	12	Base plate-2	0	−1	0	*Coincident17*
	4	Clamping plate	0	1	0	*Coincident10*		9	Set screw 1-3	0	1	0	*Coincident17*
14	2	Vice Jaw	0	−1	0	*Concentric6*	33	12	Base plate-2	0	−1	0	*Coincident18*
	5	Set screw 2-1	0	−1	0	*Concentric6*		10	Set screw 1-4	0	1	0	*Coincident18*
15	4	Clamping plate	0	−1	0	*Concentric7*	34	1	Vice Body	0	0	−1	*Parallel6*
	5	Set screw 2-1	0	−1	0	*Concentric7*		4	Clamping plate	0	0	1	*Parallel6*
16	4	Clamping plate	0	−1	0	*Coincident11*	35	1	Vice Body	0	0	1	*Parallel7*
	5	Set screw 2-1	0	1	0	*Coincident11*		4	Clamping plate	0	0	−1	*Parallel7*
17	2	Vice Jaw	0	−1	0	*Concentric9*	36	1	Vice Body	0	−1	0	*Parallel10*
	6	Set screw 2-2	0	−1	0	*Concentric9*		4	Clamping plate	0	1	0	*Parallel10*
18	4	Clamping plate	0	−1	0	*Concentric10*	37	11	Base plate-1	0	1	0	*Parallel8*
	6	Set screw 2-2	0	−1	0	*Concentric10*		4	Clamping plate	0	−1	0	*Parallel8*
19	4	Clamping plate	0	−1	0	*Coincident12*	38	12	Base plate-2	0	1	0	*Parallel9*
	6	Set screw 2-2	0	1	0	*Coincident12*		4	Clamping plate	0	−1	0	*Parallel9*

Component 1

Contact	+x	-x	+y	-y	+z	-z	Mating
1,2	1	1	1	0	1	1	*Coincident1*
1,2	1	1	1	0	1	1	*Coincident2*
1,2	1	1	1	1	1	0	*Parallel1*
1,2	1	1	1	1	0	1	*Parallel2*
1,2	0	1	1	1	1	1	*Parallel3*
1,2	1	0	1	1	1	1	*Parallel4*
1,3	1	1	0	0	0	0	*Concentric1*
1,4	1	1	1	1	0	1	*Parallel6*
1,4	1	1	1	1	1	0	*Parallel7*
1,4	1	1	0	1	1	1	*Parallel10*
1,7	0	0	1	1	0	0	*Concentric14*
1,8	0	0	1	1	0	0	*Concentric15*
1,9	0	0	1	1	0	0	*Concentric20*
1,10	0	0	1	1	0	0	*Concentric21*
1,11	1	1	0	1	1	1	*Coincident13*
1,12	1	1	0	1	1	1	*Coincident16*
AND	**0**	**0**	**0**	**0**	**0**	**0**	

Component 2

Contact	+x	-x	+y	-y	+z	-z	Mating
2,1	1	1	0	1	1	1	*Coincident1*
2,1	1	1	0	1	1	1	*Coincident2*
2,1	1	1	1	1	0	1	*Parallel1*
2,1	1	1	1	1	1	0	*Parallel2*
2,1	1	0	1	1	1	1	*Parallel3*
2,1	0	1	1	1	1	1	*Parallel4*
2,3	1	1	0	0	0	0	*Concentric2*
2,3	1	0	1	1	1	1	*Parallel5*
2,4	1	1	0	1	1	1	*Coincident10*
2,5	0	0	1	1	0	0	*Concentric6*
2,6	0	0	1	1	0	0	*Concentric9*
2,13	0	0	1	1	0	0	*Concentric3*
2,13	1	1	1	0	1	1	*Coincident9*
AND	**0**	**0**	**0**	**0**	**0**	**0**	

Component 3

Contact	+x	-x	+y	-y	+z	-z	Mating
3,1	1	1	0	0	0	0	*Concentric1*
3,2	1	1	0	0	0	0	*Concentric2*
3,2	0	1	1	1	1	1	*Parallel5*
3,13	1	0	1	1	1	1	*Tangent6*
AND	**0**	**0**	**0**	**0**	**0**	**0**	

Component 4

Contact	+x	-x	+y	-y	+z	-z	Mating
4,1	1	1	1	1	1	0	*Parallel6*
4,1	1	1	1	1	0	1	*Parallel7*
4,1	1	1	1	0	1	1	*Parallel10*
4,2	1	1	1	0	1	1	*Coincident10*
4,5	0	0	1	1	0	0	*Concentric7*
4,5	1	1	0	1	1	1	*Coincident11*
4,6	0	0	1	1	0	0	*Concentric10*
4,6	1	1	0	1	1	1	*Coincident12*
4,11	1	1	0	1	1	1	*Parallel8*
4,12	1	1	0	1	1	1	*Parallel9*
AND	**0**	**0**	**0**	**0**	**0**	**0**	

Component 5

Contact	+x	-x	+y	-y	+z	-z	Mating
5,2	0	0	1	1	0	0	*Concentric6*
5,4	0	0	1	1	0	0	*Concentric7*
5,4	1	1	1	0	1	1	*Coincident11*
AND	**0**	**0**	**1**	**0**	**0**	**0**	

Component 6

Contact	+x	-x	+y	-y	+z	-z	Mating
6,2	0	0	1	1	0	0	*Concentric9*
6,4	0	0	1	1	0	0	*Concentric10*
6,4	1	1	1	0	1	1	*Coincident12*
AND	**0**	**0**	**1**	**0**	**0**	**0**	

Component 7

Contact	+x	-x	+y	-y	+z	-z	Mating
7,1	0	0	1	1	0	0	*Concentric14*
7,11	0	0	1	1	0	0	*Concentric12*
7,11	1	1	1	0	1	1	*Coincident14*
AND	**0**	**0**	**1**	**0**	**0**	**0**	

Component 8

Contact	+x	-x	+y	-y	+z	-z	Mating
8,1	0	0	1	1	0	0	*Concentric15*
8,11	0	0	1	1	0	0	*Concentric13*
8,11	1	1	1	0	1	1	*Coincident15*
AND	**0**	**0**	**1**	**0**	**0**	**0**	

Component 9

Contact	+x	-x	+y	-y	+z	-z	Mating
9,1	0	0	1	1	0	0	*Concentric20*
9,12	0	0	1	1	0	0	*Concentric18*
9,12	1	1	1	0	1	1	*Coincident17*
AND	**0**	**0**	**1**	**0**	**0**	**0**	

Component 10

Contact	+x	-x	+y	-y	+z	-z	Mating
10,1	0	0	1	1	0	0	*Concentric21*
10,12	0	0	1	1	0	0	*Concentric19*
10,12	1	1	1	0	1	1	*Coincident18*
AND	**0**	**0**	**1**	**0**	**0**	**0**	

Component 11

Contact	+x	-x	+y	-y	+z	-z	Mating
11,1	1	1	1	0	1	1	*Coincident13*
11,4	1	1	1	0	1	1	*Parallel8*
11,7	0	0	1	1	0	0	*Concentric12*
11,7	1	1	0	1	1	1	*Coincident14*
11,8	0	0	1	1	0	0	*Concentric13*
11,8	1	1	0	1	1	1	*Coincident15*
AND	**0**	**0**	**0**	**0**	**0**	**0**	

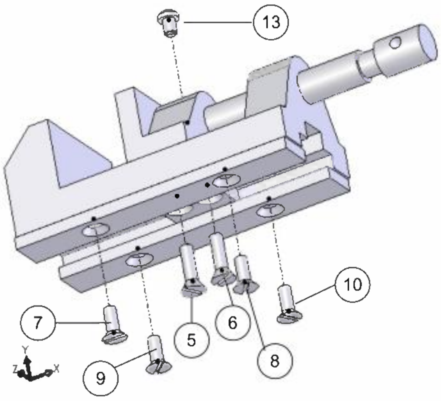
Component 12

Contact	+x	-x	+y	-y	+z	-z	Mating
12,1	1	1	1	0	1	1	*Coincident16*
12,4	1	1	1	0	1	1	*Parallel9*
12,9	0	0	1	1	0	0	*Concentric18*
12,9	1	1	0	1	1	1	*Coincident17*
12,10	0	0	1	1	0	0	*Concentric19*
12,10	1	1	0	1	1	1	*Coincident18*
AND	**0**	**0**	**0**	**0**	**0**	**0**	

Component 13

Contact	+x	-x	+y	-y	+z	-z	Mating
13,2	0	0	1	1	0	0	*Concentric3*
13,2	1	1	0	1	1	1	*Coincident9*
13,3	0	1	1	1	1	1	*Tangent6*
AND	**0**	**0**	**0**	**1**	**0**	**0**	

Fig. 5. Collision-free path of assembly for each components of Bench Vice.



No	Component	Assembly path
5	Set screw 2.1	$+y$
6	Set screw 2.2	$+y$
7	Set screw 1.1	$+y$
8	Set screw 1.3	$+y$
9	Set screw 1.2	$+y$
10	Set screw 1.4	$+y$
13	Oval fillister	$-y$

Fig. 6. Collision-free path of a number of Bench Vice components.

The collision-free path of assembly for all components is evaluated using AND logic. The result of collision-free path of assembly for all Bench Vice components using Collision Detection Algorithm is shown in Fig. 5.

Based on the information of collision-free path of assembly, a number of directions can be used to assemble each component according to the six-axis. Component 5, 6, 7, 8, 9, and 10 can be assembled in $+y$ direction, while the component 13 can be assembled into $-y$ direction (see Fig. 6). The information of collision-free path of assembly is used to establish precedence constraint using disassembly approach. The information of collision-free path of assembly reevaluated every component that was disassembled. The use of disassembly approach must be followed by a change in direction signs on collision-free path of assembly. Collision-free path of assembly shown in Fig. 6 is the first step to generate precedence constraints of Bench Vice. Figure 6 shows the components that can be disassembled, which are components 5, 6, 7, 8, 9, 10 and 13 in the opposite way to the assembly direction. Automation of collision detection analysis, and precedence constraints generation, are part of an effort to develop an automated method of assembly sequence generation, which is currently being developed.

5. Conclusion

This paper discussed the proposed method for detecting the collision-free path of a component assembly based on geometric information from CAD systems database. The method is fully automated using 3D solid drawing as input. The collision-free path of assembly is determined based on mating type, the normal vector direction, and the surface shape of contacted component. This method is able to detect collision between components that have physical contact (local collision) or without physical contact (global collision). This study is a part of an effort to develop an automated method in generating precedence constraint which is currrently being developed.

This research has succeeded in simplifying the automated method of detection the collision-free path of assembly. The collision-free path of assembly can be defined directly based on the information of contacted components, mating type, and direction of normal vector of contacted face of components. To build the proposed method, three propositions has been established and proven, one set of rules and two algorithms have been developed and tested. For implementing this algorithm, a module of software prototype has been produced.

Appendix A. Component Geometry Database Formation

Component geometrical data required for the collision detection algorithm taken from the CAD database of product, the data stored in The Component Geometry Database and The Collision-Free Path of Assembly Database. The data extraction, the formation of The Component Geometry Database and The Collision-Free Path of Assembly Database, conducted by algorithms developed in this paper. Figure A.1 shows the diagram of the developed system.

Component Geometry Database is prepared to provide all data required in analysis of collision-free path of assembly. The data extracted automatically from the CAD system database using 3D stacked drawing of products in *SolidWorks* 2005 as input. The data extraction explained in The Algorithm of Database Formation of Component Geometry. Graph model of Alfadhlani and Toha[13] adopted and modified to explain this algorithm, see Fig. A.2.
where:

p_k is the kth component.
k is the index components were observed, $k = 1, 2, \ldots, N$; $k \neq 1$.
l is the index components are connected with p_k, $l = 1, 2, \ldots, N$; $l \neq k$.
v_k is volume of the kth component.
$\mathbf{a}_k$ is the normal vector of face the kth component in contacts.
m_{kl} is a mating occurred between component p_k and component p_l.

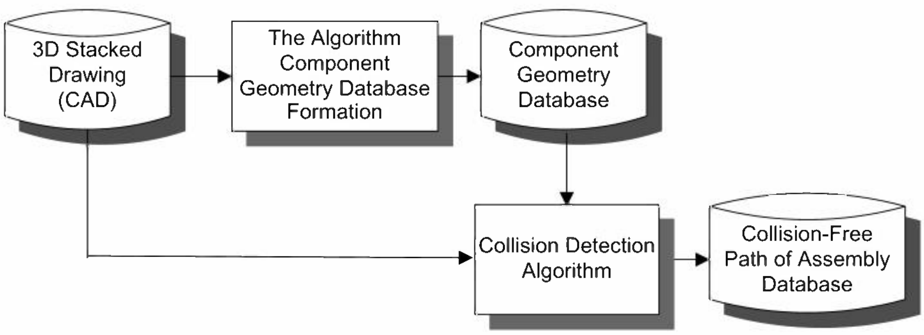


Fig. A.1. Developed system diagram.

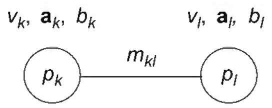


Fig. A.2. Graph model of assembly.

The algorithm of database formation of component geometry:

The algorithm for extracting data from CAD database and storing in The Component Geometry Database is as follows:

(1) Make sure the *SolidWorks* assembly document is active.
(2) Get the mating name between pairs of components that contact, (mating information taken from the active drawing in *SolidWorks* with the property *Name* of the object class *Feature*).
(3) Check the mating type of the pair contacted components, if the mating type is *coincident*, *parallel*, *tangent* or *distance*, then enter the next step, otherwise return to step 2.
(4) Get one name of the component from the pair of contacts (the component name is taken with property *Name* on an object *Component2*).
(5) Get its volume ($v_i, i = 1, 2, \ldots, N$) (volume information is taken by using the property *Volume* on the object class *MassProperty*).
(6) Identification of the normal vector direction for each face of component that contacts ($\mathbf{a}_k$) (this information is taken from the active drawing of *SolidWorks* with property *EntityParams* (l) on the object class *MateEntity2*, l is the directions $i - j - k$ where the value $l = 3, 4, 5$). If there is a normal vector that does not fit with one of the axes x, y, or z, then stop, the iteration is complete.
(7) Find the location of its mating in assembly-model-space (b_k) (this information is taken from the active drawing of *SolidWorks* with property *EntityParams* (l) on the object class *MateEntity2*, l is the directions $x - y - z$ where the value $l = 0, 1, 2$).
(8) Save the information obtained from step 3 to step 7 on component geometry database ($p_k, v_k, \mathbf{a}_k, b_k$). If it's mate component has not been checked, then go back to step 4. Otherwise, proceed to next step.
(9) Check again the mating type of the pair of contacted component which has just been checked, if the type of mating is *coincident* then proceed to the next step, if *tangent* or *distance* go to step 12, the others go to step 13.
(10) Check the face shape of the two components that contacts, if both are *cone* shaped then check the direction of its mating alignment. Information of face shape that contacts obtained by using the property *ReferenceType* on the class object *MateEntity2*, mating alignment information obtained by using the property *Alignment* on the object class *Mate2*.

(11) If the mating alignment is to align, then compare the volume of both components, change the direction of normal vector component that has the greatest volume into the opposite direction to the normal vector at the moment, and go to step 13. Otherwise go directly to step 13.
(12) Check the face shape of the two components that are in contact; if the pair face that contact is the *plane* with a *cylinder* or a *plane* with a *sphere*, then change the direction of normal vectors face shaped *cylinder* or *sphere* in opposite directions with the normal vector face shaped *plane*.
(13) If there are any mating which are not checked then check the next mating and return to step 4, otherwise the iteration is complete.

The database of component geometry is formed, containing information about a list of all pairs of contacted components and their mating type, the volume of each component, the coordinates of the point of contact, and their normal vector directions.

References

1. G. Boothroyd, P. Dewhurst and W. Knight, *Product Design for Manufacture and Assembly*, (Marcel Dekker, New York, 1994).
2. A. Kusiak, *Intelligent Manufacturing Systems*, (Prentice Hall, New Jersey, 1990).
3. D. E. Whitney, *Mechanical Assemblies: Their Design, Manufacture, and Role in Product Development*, (Oxford University Press, 2004).
4. E. Kroll, Modeling and reasoning for computer-based assembly planning, in *Concurrent Engineering: Automation, Tools, and Techniques*, ed. A. Kusiak (1992) 177–205.
5. H. Y. Lai and C. T. Huang, A systematic approach for automatic assembly sequence plan generation, *International Journal of Advance Manufacturing Technology* **24** (2004) 752–763.
6. A. Delchambre, *Computer-Aided Assembly Planning* (Chapman & Hall, London, 1992).
7. T. L. De Fazio and D. E. Whitney, Simplified generation of all mechanical assembly sequences, *IEEE Transactions on Robotics and Automation* RA-**3**(6) (1987) 640–658.
8. P. Gu and X. Yan, CAD-Directed automatic assembly sequence planning, *International Journal of Production Research* **33**(11) (1995) 3069–3100.
9. S. S. F. Smith, G. C. Smith and X. Liao, Automatic stable assembly sequence generation and evaluation, *Journal of Manufacturing System* **20**(4) (2001) 225–235.
10. C. Pan, S. S. Smith and G. C. Smith, Automatic assembly sequence planning from STEP CAD files, *International Journal of Computer Integrated Manufacturing* **19**(8) (2006) 775–783.
11. R. Ariastuti, H. Taroepratjeka and I. S. Toha, Automatic assembly sequence generation of products, *Jurnal Teknik dan Manajemen Industri* **18**(1) (1998), (in Indonesian).
12. S. Tickoo, *SolidWorks for Designers Release 2004*, (CADCIM Technologies, Schererville, USA, 2004).
13. Alfadhlani and I. S. Toha, Product assembly sequence generation using 3D solid drawing as input, *Proceedings of Seminar VII Production System* (2005) 1–13, (in Indonesian).