Generation of Disassembly Directions Based on Component Mobility

Robert IACOB, Diana POPESCU
University Politehnica of Bucharest,
313, Splaiul Independentei, 060032 Bucharest 6, Romania,
robert.iacob@gmail.com, diana@mix.mmi.pub.ro

Abstract: The paper presents the development of a new software tool integrated with SolidWorks CAD system for generating feasible disassembly directions based on the concept of mobility operator. In generating the disassembly path of a component, our approach considers that two steps should be undertaken: first, the interface between components is broken and then the component is extracted from the assembly by translating it along a collision free removal path. Different data from the CAD model of the assembly is automatically detected, such as: geometrical constraints, contact surfaces relative position, common area, components neighbours, etc. Based on this information, a general mobility operator is deployed, which is further used to evaluate the families of trajectories associated to the contacts between different components of a mechanical system. Thus, the developed mobility operator mathematically expresses all possible movements between assembly components (rotations, translations and helical movements), and it can be used for a real time graphical representation of components mobilities, mandatory aspects in developing a virtual environment for simulating the Assembly/Disassembly (A/D) tasks as part of an efficient Product Design Process (PDP).

Keywords: assembly/disassembly, disassembly planning, geometrical constraint, mobility operator

1. Introduction

Although the paradigm of Environmental Conscious Manufacturing (ECM) is not new [Gungor, 1999], [Mascle, 2008], the last couple of years showed an increasing interest and concern in designing and manufacturing products whose components can be recycled or reused in a high percentage. This is due to the reinforcement of European directives such as WEEE 2002/96/EC on Waste Electrical and Electronic Equipment or 2000/53/EC on end-of-life vehicles, which extends producers responsibility on the entire life cycle of a product, therefore also over the components/materials re-using or recycling activities. Products should be designed to be disassembled at the end of their life cycle and parts of them to be used in generating other products, by refurbishing, reuse, recycling or to be safely disposed, in case of dangerous materials/components.

Therefore, in this competitive world and in the global context of concern over the environment and resources, the producers need to reconsider the design process of a product as part of a larger perspective which includes a correct assessment of the impact of different design decisions on the post-consumer stage. In this sense, every bad design decision can cause a reduction on the producer market share and even to its economic failure. Solutions should be found for satisfying client requirements, ensuring profitability for producer and complying with regulations and laws, all in the same time. Thus, there is a need of tools – here including software applications – which can help engineers to take well documented design decisions when modeling a product, leading to a decrease of the costs and time associated to assembly, repair, disassembly or recycling operations.

In this context, the current research proposes an intelligent software tool for assisting the disassembly planning. This tool is integrated with SolidWorks CAD system for generating feasible dismounting plans and it is based on a mobility operator formed by two elements: the mathematical description of the general movements of a component and the real time graphical representation of the component mobility relative to its neighbours.

The proposed software application can assist engineers in establishing valid disassembly plans based on the 3D CAD representation of the product. Having, from the design phase of a product, the complete information of the disassembly operations and components valid movements can be useful in the early determination of the architecture of a mechanism, machine, robot or disassembly tool adapted to perform the operations in question, and also for estimating disassembly costs and time.

From this point forward, the paper is organized as follows. Chapter 2 presents a literature study grouped in two major categories: researches focused on using unit sphere for mapping valid disassembly directions and software applications for disassembly sequence planning. Our approach extends the existing concepts in order to include all the possible movements. Moreover, as another novelty, the software application integrates the mobility operator with the 3D CAD representation of the product for generating feasible disassembly directions. Chapter 3 is focused on defining the mobility operator for rotations, translations and helical movements from two points of view: mathematical and geometrical. Chapter 4 presents the development of the software application, describing its data structure and information flow, the pseudo-code for generating the mobility operator implemented in the application, as well as a case study. Chapter 5 presents the conclusions of the paper and further research work.

2. Literature Review

Disassembly is an activity as old as the first products ever manufactured, being performed for maintenance, repair, reuse, and, more recently, for recycling some parts or the entire assembly/product. The interest in studying disassembly is focused on finding the optimal way to perform it, that is the easiest (in term of access to different parts, number of tools and operations required, number of repositions of products during disassembly etc.), as well as the cheapest, in terms on time and cost (labour, tools, etc.). However, determining the optimal disassembly sequence is not an easy task, the number of possible disassembly sequences exponentially growing with the number of components in an assembly. Therefore, the researchers are proposing different strategies to reduce this computational effort. Literature reviews are comparatively presenting and classifying these strategies [Santochi, 2002], [Lambert, 2003], [Dong, 2003], [Ilgin, 2010], [Seth, 2011], [Jimez, 2012], [Iacob, 2012], and represent the results of investigating hundreds of pages of studies in this field.

The advances in the computers and programming determined the researchers to focus their attention on finding approaches [Duta, 2008] and on developing different models for automating the generation of disassembly sequences and plans. However, the number of studies presenting the software applications which actually implements these approaches is significantly lower than the papers presenting concepts and frameworks, and can be divided in two broad categories: CAD based disassembly planning software and Virtual Reality (VR) based disassembly planning. This paper presents a software application for the automatic generation of feasible disassembly directions which implements the theoretical concept of mobility operator, as a general expression of the valid families of trajectories associated to the contacts between different components of a mechanical system. Hence, the following literature review is focused on studies which describe CAD and VR based disassembly software applications, with a special consideration to those using the unit sphere concept for determining the escape direction of a component from the assembly.

Mo et al. present the implementation of a Virtual Disassembly Analyzer (VDA), which generates and evaluates disassembly removal directions based on the CAD models (B-rep representations), for improving the design of a product [Mo, 2002]. The application calculates the disassembly directionality of a component using contact constraints and Gaussian sphere concept. The Gaussian sphere is a normal map to a geometric entity defined on the unit sphere – the set of points of distance 1 from a fixed central point. The interference between components, when removed from the assembly, is calculated using the bounding boxes, while the sequence generation is based on the concept of propagation waves.

In [Pomares, 2004], the CAD model of products (converted as a VRML model) and the corresponding list of components features are used for developing local and global strategies for component removal. Common types of contacts between components are considered, such as screw and cylindrical, this data being manually provided by user. Gaussian sphere is used for determining the direction of disassembly for a component, while the directions of removal are determined based on the collisions detected between polyhedral objects in motion along a rectilinear trajectory using linear programming. The proposed method is implemented in a simulation system allowing the evaluation of different disassembly movements.

Puente et al. use the hierarchical model of the product to represent the relations among components in order to plan the disassembly operations [Puente, 2010]. The presented algorithm, which is an extension of the previous mentioned work, computes the best sequence of components to remove a target material instead of a target component. The information is used for programming and distributing the tasks between robots which should perform disassembly in a collaborative manner.

Li et al. developed an object-oriented disassembly planner for maintenance operations based on the constraint graph representation scheme [Li, 2005], [Li, 2006]. The generation of near optimal disassembly sequences is made using genetic algorithms. The developed prototype software consists in three modules, the planner module generating the optimal disassembly sequence using a graphical user interface. A target component is chosen by the user, the planner displaying the possible disassembly sequences. The focus of this research is on finding the direction of each disassembly operation and the tools required for performing it in order to optimize de disassembly cost. Although based on the constraint graph, these studies are interesting for our research from the software application architecture point of view.

Chung and Peng present the implementation, in software application, of a method for generating a feasible disassembly plan based on the constraints of a given part position and the fastener type, provided by user [Chung, 2006].

The authors developed a digitized assembly directionality chart (D^3C) based on the concept of discrete global accessibility cones, as a representation of the surroundings of a component targeted for disassembly.

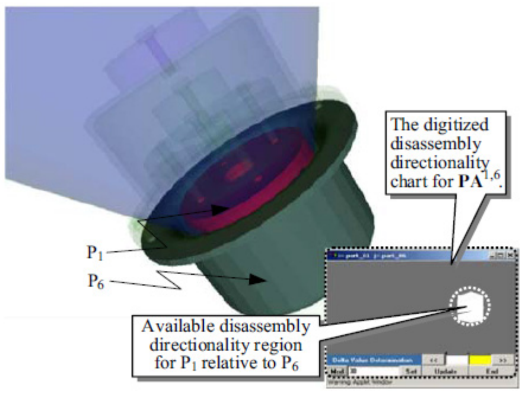

The digitized disassembly directionality chart for $PA^{1,6}$.
P_1
P_6
Available disassembly directionality region for P_1 relative to P_6

Figure 1. Disassembly directionality check. [Chung, 2006]

The available removal directions of a component relative to a neighbouring component is determined mapping the non-accessible direction region on D^3C of each parts after automatically detecting two-contacted triangle patches of these. Thus, the simulation software developed can clearly verify whether the process is appropriate or not for a fastener disassembling (Figure 1).

Aguinaga et al. propose a VR disassembly planner which implements a two paths planning technique for dismounting for aircraft engines and equipment, one based on single translations and the other based on random tree search [Aguinaga, 2007]. The implemented algorithm detects first the contacts between components, generates disassembly sequence and then determines the removal directions using the discrete representation of the Gaussian sphere. This approach is very interesting, not requiring user input other that the selection of the component for disassembly, but only translation movements are considered for dismounting.

Takeuchi and Saitou describe a “domino-like disassembly” approach based on components geometry, on a locator library which is used for defining feasible movement directions and on a defined priority set relating to disassembly cost and components recycling revenues [Takeuchi, 2008]. A study case for removing fasteners is used for explaining how “Pareto” optimal disassembly solutions are found based on a multi-objective genetic algorithm. These methods can be used with success if the configuration space is not too narrow; however, a higher level of automation would be required.

Smith and Chen present a disassembly approached based on the geometric relationships between components, rather than geometric constraints for pairs, the proposed algorithm being recursive and based on five rules defined to eliminate unfeasible or unrealistic sequences [Smith, 2011]. Therefore, not all the possible disassembly solutions are generated, the optimal disassembly path being evaluated using a cost function. The escape directions of a component from an assembly are calculated using constraint matrices, which are corroborated with the mating features determined from the CAD model of the product. The implemented algorithm uses the distinction between components and screws, but this input is provided manually by the user along with the constraints in the first steps.

Different scenarios and methods are deployed when the Assembly/Disassembly (A/D) trajectories are simulated in a real-time environment using haptic devices [Perret, 2013]. A haptic device, also called a force-feedback device, is a computer peripheral, which can apply forces to the hand of its user. Haptic interaction is highly bidirectional: the user inputs information into the system through the movement of the haptic device end effector, and, at the same time, the user feels the system output as a force. Interactive simulation with haptic feedback recreates a physical interaction with the 3D model, calling upon the manual skills and cognitive capabilities of the human operator. By performing the assembly operations "hands-on", the design engineer or the process planner measures the complexity and evaluates also the ergonomic dimension of the tasks (Figure 2).

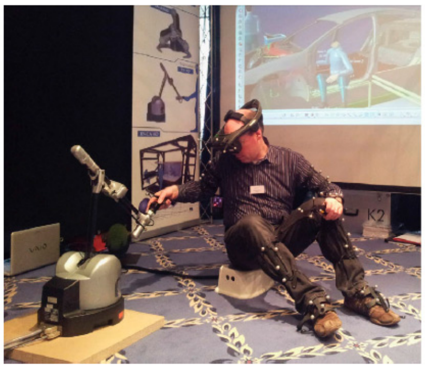


Figure 2. Simulation of an assembly operation. [Perret, 2013]

The analysis of the above mentioned studies showed that their common limitations are related mainly to an incomplete automation of the disassembly plans generation. This is due to the fact that the components cannot be identified based on their functional role and that the valid disassembly trajectories do not include helical movements, but mostly translations, and seldom rotations. Moreover, the CAD models used as input geometry are based on triangular mesh which influences the correct identification of contact surfaces between components. Therefore, there is a need of processing the CAD model prior to the contact identification, step that is not considered in the analysed literature studies.

In this context, our research aims at developing an application for automatically detecting disassembly directions using 3D CAD data and calculating components mobility with respect to its surrounding components.

This is part of a larger project called ViPAD – Virtual Platform for A/D Simulation, which considers the generation and simulation of the valid A/D trajectories of components from a mechanical assembly.

3. Generation and Representation of the Disassembly Directions

In order to model the contact relations between two components of an assembly, there is a need to explicitly describe their relative mobility, i.e. reciprocal authorized displacements. All possible disassembly trajectories (directions) for each component should be analyzed, for the three basic type of movements, in order to find the proper ones, which leads to the specification of an operator. Hence, this mathematical model expresses all the families of trajectories associated to the contacts between different components of a product, which corroborated with path finding algorithms should generate feasible, collision free dismounting trajectories.

For characterizing the family of trajectories resulting from the combination of two different contacts: C_1 and C_2, each elementary contact between the reference component and the surrounding ones will be associated to a bi-quaternion. The dual quaternions offer the advantage of representing, in a unified manner, different possible movements (rotations, translations, helical movements), the result defining the possible/feasible motions of the reference component with respect to the whole set of contacts (elements) considered for the analysis [Iacob, 2011].

In the following sections an analysis of the three basic types of movements: rotation, translation and helical, is presented in order to characterize the compatibility of trajectories for each type of trajectory, and then the way in which they are represented and combined. For each type of movement, the mathematical model and the geometrical representation will be detailed for defining the mobility operator.

3.1 Rotation movements

Rotation movements can be expressed using standard or dual quaternions, these two descriptions being equivalent. Although the proposed operator is based on dual quaternions, we start by presenting how the rotations are expressed using standard quaternions, and then the transformation in dual quaternions.

The rotation transformation in space of a vector $\vec{u} = (X, Y, Z)$ with its corresponding quaternion $\vec{u} = \{o, \vec{u}\}$ can be described as presented in (1) using standard quaternions.

$$U = Q \cdot u \cdot Q^* \qquad (1)$$

The parameters in equation (1) are the following:

$$U = \{0, (U_X, U_Y, U_Z)\},\ u = \{0, (u_X, u_Y, u_Z)\} \qquad (2)$$

$$Q = \left\{\cos\frac{\theta}{2}, \left(\sin\frac{\theta}{2} \cdot v_X, \sin\frac{\theta}{2} \cdot v_Y, \sin\frac{\theta}{2} \cdot v_Z\right)\right\} \qquad (3)$$

$$Q^* = \left\{\cos\frac{\theta}{2}, \left(-\sin\frac{\theta}{2} \cdot v_X, -\sin\frac{\theta}{2} \cdot v_Y, -\sin\frac{\theta}{2} \cdot v_Z\right)\right\} \qquad (4)$$

Where Q and Q^* are written in the scalar-vectorial form: $Q = \{S, \vec{V}\}$, $Q^* = \{S, -\vec{V}\}$.

The representation of a transformation using dual quaternions is based on the general transformation in space of a dual vector, as defined in equation (5):

$$\hat{U} = \hat{Q}_S \cdot \hat{u} \cdot \hat{Q}_D \qquad (5)$$

The dual representation of the two vectors in equation (5) is:

$$\hat{U} = \{1 + \varepsilon \cdot U\},\ \ U = \{0, (U_X, U_Y, U_Z)\} \qquad (6)$$

$$\hat{u} = \{1 + \varepsilon \cdot u\},\ \ u = \{0, (u_X, u_Y, u_Z)\} \qquad (7)$$

The dual quaternions $\hat{Q}_S$ and $\hat{Q}_D$, for a pure rotation, have the following expressions:

$$\hat{Q}_S = \{Q_R + \varepsilon \cdot 0\} \qquad (8)$$

where

$$Q_R = \left\{\cos\frac{\theta}{2}, \left(\sin\frac{\theta}{2} \cdot v_X, \sin\frac{\theta}{2} \cdot v_Y, \sin\frac{\theta}{2} \cdot v_Z\right)\right\} \qquad (9)$$

$$\hat{Q}_D = \{Q_R^* - \varepsilon \cdot 0\} \qquad (10)$$

where

$$Q_R^* = \left\{\cos\frac{\theta}{2}, \left(-\sin\frac{\theta}{2} \cdot v_X, -\sin\frac{\theta}{2} \cdot v_Y, -\sin\frac{\theta}{2} \cdot v_Z\right)\right\} \qquad (11)$$

Thus, the transformation becomes:

$$\begin{aligned}\hat{U} &= \hat{Q}_S \cdot \hat{u} \cdot \hat{Q}_D = \{Q_R + \varepsilon \cdot 0\} \cdot \{1 + \varepsilon \cdot u\} \cdot \{Q_R^* - \varepsilon \cdot 0\} = \\ &= \{Q_R \cdot Q_R^* + \varepsilon \cdot Q_R \cdot u \cdot Q_R^*\} = \{1 + \varepsilon \cdot Q_R \cdot u \cdot Q_R^*\}\end{aligned} \qquad (12)$$

Based on quaternions properties: $Q_R \cdot Q_R^* = 1$, therefore:

$$U = Q_R \cdot u \cdot Q_R^* \qquad (13)$$

which represent a transformation equivalent with the rotation presented at the beginning of this section.

The rotation movements are represented on a unit sphere S_U, $\{O, \vec{x}, \vec{y}, \vec{z}\}$, with the property that: $\forall M \in S_U$, $\left\|\overrightarrow{OM}\right\| = 1$, in accordance with Figure 3. The vector $\overrightarrow{OM}$ defines a rotation axis. The positive direction of the axis represents a possible (valid) counter-clockwise rotation described by a unit quaternion:

$$Q = \left\{\cos\frac{\theta}{2}, \left(\sin\frac{\theta}{2}\right)\overrightarrow{OM}\right\},\ \theta > 0 \qquad (14)$$

The negative direction represents a possible (valid) clockwise rotation about the same axis. Such a rotation is described by:

$$Q' = \left\{\cos\frac{\theta'}{2}, \left(\sin\frac{\theta'}{2}\right)\overrightarrow{OM'}\right\}, \theta' > 0, \overrightarrow{OM} = -\overrightarrow{OM'} \qquad (15)$$

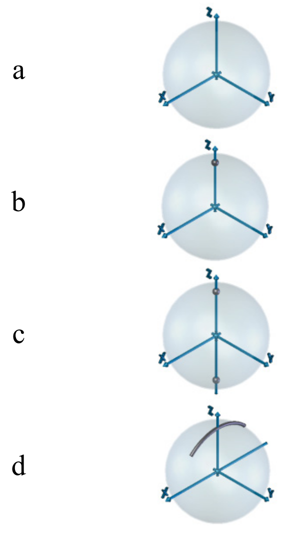


Figure 3. Rotation movements' representation.

Figure 3 contains some examples of rotation movements' representation on the unit sphere. Each case describes a family of trajectories:

a – no possible movement;

b – one rotation is possible: counter-clockwise rotation around the Z axis;

c – two rotations are possible: rotation in both directions (positive and negative) around the same axis Z;

d – the valid rotation directions represent a circular sector in the plane (X, Z).

Using this geometrical method all the rotation movements, associated with a contact, can be represented by a combination of geometrical elements on a unit sphere.

3.2 Translation movements

The translation movements can be completely represented only using dual quaternions. Therefore, for a pure translation equation (5) is used, the dual quaternions $\hat{Q}_S$ and $\hat{Q}_D$ having the following expressions:

$$\hat{Q}_S = \left\{1 + \varepsilon \frac{Q_T}{2}\right\},\ Q_T = \{0, (t_X, t_Y, t_Z)\} \tag{16}$$

$$\hat{Q}_D = \left\{1 - \varepsilon \frac{Q_T^*}{2}\right\},\ Q_T^* = \{0, (-t_X, -t_Y, -t_Z)\} \tag{17}$$

The transformation becomes:

$$\begin{gathered} \hat{U} = \hat{Q}_S \cdot \hat{u} \cdot \hat{Q}_D = \left\{1 + \varepsilon \frac{Q_T}{2}\right\} \cdot \{1 + \varepsilon \cdot u\} \cdot \left\{1 - \varepsilon \frac{Q_T^*}{2}\right\} \\ \hat{U} = \left\{1 + \varepsilon \cdot \left(\frac{Q_T}{2} + u - \frac{Q_T^*}{2}\right)\right\} \end{gathered} \tag{18}$$

Based on quaternions properties: $Q_T = -Q_T^*$, therefore: $\hat{U} = \{1 + \varepsilon \cdot (u + Q_T)\}$. In this case, the new vector is:

$$U = u + Q_T \tag{19}$$

Translation movements can be represented on a unit sphere S_U as indicated in the previous section. The representation is similar to the rotations, but with a different meaning: the vector $\overrightarrow{OM}$ defines a translation axis. The positive direction of one axis represents a possible (valid) translation:

$$\vec{T} = \lambda \cdot \overrightarrow{OM},\ \lambda > 0 \tag{20}$$

The negative direction represents a possible (valid) translation about the same axis, but opposite. Such a translation is described by:

$$\vec{T}' = \lambda' \cdot \overrightarrow{OM}',\ \lambda' > 0,\ \overrightarrow{OM} = -\overrightarrow{OM}' \tag{21}$$

Figure 4 contains some examples of translation movements' representation on the unit sphere:

a – one translation is possible following the Z axis: positive;

b – two translations are possible following the Z axis: positive and negative;

c – the valid translation directions represent a circular sector in the plane (X, Z);

d – the valid translation directions represent a sector of the unit sphere.

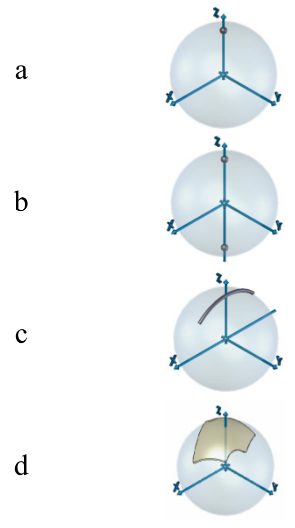


Figure 4. Translation movements' representation.

3.3 Helical movements

Knowing that "the most general rigid body displacement can be produced by a translation along a line followed (or preceded) by a rotation about the same line" [Chasles, 1830], the same equation (5) will be used. In this case, the dual quaternions $\hat{Q}_S$, $\hat{Q}_D$ are:

$$\hat{Q}_S = \left\{Q_R + \varepsilon \frac{Q_T \cdot Q_R}{2}\right\} \tag{22}$$

where Q_R and Q_T are the quaternions for rotation and translation;

$$\hat{Q}_D = \left\{Q_R^* - \varepsilon \frac{Q_R^* \cdot Q_T^*}{2}\right\} \tag{23}$$

where Q_R^* and Q_T^* are the conjugated quaternions of Q_R and Q_T.

Thus, the general transformation becomes:

$$\begin{gathered} \hat{U} = \hat{Q}_S \cdot \hat{u} \cdot \hat{Q}_D \\ \hat{U} = \left\{Q_R + \varepsilon \frac{Q_T \cdot Q_R}{2}\right\} \cdot \{1 + \varepsilon \cdot u\} \cdot \left\{Q_R^* - \varepsilon \frac{Q_R^* \cdot Q_T^*}{2}\right\} \\ \hat{U} = \left\{Q_R \cdot Q_R^* + \varepsilon \cdot \left(Q_R \cdot u \cdot Q_R^* + \frac{Q_T \cdot Q_R \cdot Q_R^*}{2} - \frac{Q_R \cdot Q_R^* \cdot Q_T^*}{2}\right)\right\} \end{gathered} \tag{24}$$

Based on quaternions properties, it results that:

$Q_R \cdot Q_R^* = 1$ and $Q_T = -Q_T^*$, therefore: $\hat{U} = \{1 + \varepsilon \cdot (Q_R \cdot u \cdot Q_R^* + Q_T)\}$ and the new vector is:

$$U = Q_R \cdot u \cdot Q_R^* + Q_T \tag{25}$$

For this category of movements, the unit sphere S_U is no longer usable because it is necessary to represent simultaneously an axis of translation, the associated rotation and the relation between rotation and translation, which is the characteristic of a helical motion. It should be noted that the translational movements coincide with the helical movements when the pitch is infinite. Similarly, helical movements coincide with the rotational movements when the pitch is zero. Therefore, the sphere S_U, which has only two independent parameters to locate the direction of movement, is insufficient because it cannot characterize the pitch of the movement.

Consequently, the representation is different from those used for the translations and rotations. In this context, a new type of representation, that combines the movements of translation and rotation, is proposed: the unit ball. This one will be later used to develop the mobility operator.

Compared to the unit sphere S_U, the unit ball B_U defines a volume V, $\{O,\vec{x},\vec{y},\vec{z}\}$, with the property that: $\forall P \in B_U$, $\left\| \overrightarrow{OP} \right\| \leq 1$.

A point P in this volume characterises the helical movement as follows: the vector defined by the origin O and the point P represents the direction of the helical movement, the distance from P to its origin defines the pitch p, $0 \leq p \leq +\infty$. It varies between 0 for rotation, when $P \equiv O$, and $+\infty$ for translation, when $P \in S_U$, the unit sphere S_U being the border of the unit ball B_U. The corresponding rotation is defined by:

$$Q = \left\{ \cos\frac{\theta}{2}, \left(\sin\frac{\theta}{2} \right) \cdot \frac{OP}{\|OP\|} \right\},\ \theta > 0 \tag{26}$$

The dual part of the dual quaternion $\hat{Q}$ is:

$$Q = \left\{ -\frac{p}{2} \cdot \sin\frac{\theta}{2}, \frac{p}{2} \cdot \cos\frac{\theta}{2} \cdot \frac{OP}{\|OP\|} \right\} \tag{27}$$

The dual quaternion:

$$\hat{Q} = Q + \varepsilon \cdot Q_0 \tag{28}$$

represents the complete description of any type of movement.

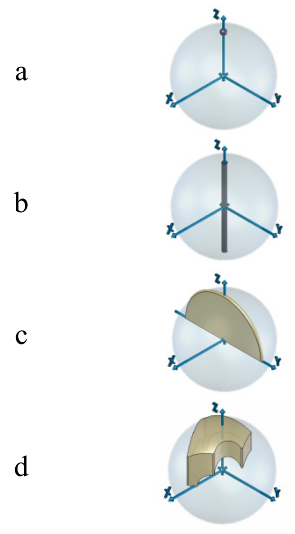


Figure 5. Helical movements' representation.

Some examples of helical movements are represented, using the ball unit, in Figure 5:

a – only one translation is possible following the Z axis: positive;

b – the valid helical movements are directed along the Z axis: they are a family of helical movements ranging between rotations (pitch zero) to translations (infinite pitch);

c – the valid helical movements represent a circular sector of B_U: all directions of possible movements are directed along axes from the plane (XZ);

d – the helical movements represent a volume of the B_U: the directions of translation or the rotation axes follow a combination between X, Y, Z, such that the Z component is always positive and the pitch p of these movements is strictly positive and reaches $+\infty$ in order to represent translational movements.

3.4 Mobility operator

As presented in the previous sections, the dual quaternions, the unit sphere concept S_U and the unit ball concept B_U are powerful tools able to describe and to represent any family of trajectories (set of movements). However, for some applications, a compact representation of the mobility would be an advantage. In addition, another important objective of the present research is to provide a combination method of different families of trajectories (directions).

Based on an extended analysis of the compatibility of trajectories [Iacob, 2011], a set of conclusions, for the three types of movements, can be drawn:

– Rotations – the movements are compatible if the axes are coaxial and the origins of the rotation vectors are coincident or shifted with respect to each other along their common direction.

– Translations – the movements are compatible if they follow the same direction and have identical absolute values.

– Helical movements – in order to have a compatibility between two helical movements, the two vectors defining the axes movements must be coaxial; in addition, the 'speeds' attached to the first and to the second element, i.e. the tangent to trajectories compatible with the first and the second element, must be the same and the pitches must be equal as well. Also, the rotation angles can be shifted with respect to each other with a fixed value.

Using all these information, a compact mobility operator for the disassembly directions representation is introduced. Thus, a family of trajectories – a set of disassembly directions, is represented using a volumetric domain D_D on a unit ball B_U (Figure 6). The F_T face is coincident with the S_T – sphere of translations, the outer surface of B_U, which correspond to the helical movements with an infinite pitch $p \to +\infty$. The helices with $p \to 0$ are situated on the S_R – sphere of rotations ($r \to 0$), close to the origin of B_U. The corresponding face of the volumetric domain is F_R. Therefore, the lateral face F_H of D_D is defined by a set of vectors representing the directions of helical movements and is located between F_R and F_T.

In order to completely define the representation from Figure 6, the following observations are necessary:

– The two spheres: S_R and S_T share the same origin with the B_U.

– The directions of rotation or translation are identical with the helical movements.

– The volumetric domain D_D is bounded by three surfaces: F_R, F_H, F_T.

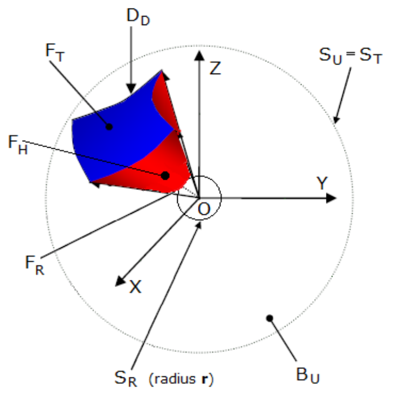


Figure 6. Disassembly directions representation.

The proposed operator can be used to represent the mobility of a single element (contact, component etc.) or to compute the mobility of a set of components (sub-assembly or assembly). For the moment, the operator is deployed for standard surfaces, but a generalization of the proposed method is considered by including the general surfaces. Thus, using the developed methodology, a mathematical description and a geometrical representation of any type of movement can be defined and then integrated in an algorithm for the real-time simulation of A/D paths.

4. Software Tool for Component Mobility Generation

The literature review conducted in Chapter 2 shows that the A/D analysis and simulation process covers a range of areas, one of the most important being the trajectory planning of components in a complex environment, which relies heavily on the 3D shape representation of the assembly. In addition, it should be noted that the relative mobility of the components is a key element that contributes to the A/D simulation and represents mandatory information.

The above characteristics are elements of an A/D application and represent a set of functions used for the analysis of mechanical systems, thus emphasizing certain conditions necessary to provide a more generic platform and a better integration of the simulation stage in the PDP.

4.1 Software architecture and data flow

The input geometry for the method presented in this paper is based on the STEP file of the assembly due to the advantages offered by the B-Rep NURBS description in terms of a more transparent access to the behaviours of the assembly components during different steps of the A/D simulation.

However, in order to be used, the CAD models should be pre-processed. This step is required because of the topological conditions specific to all CAD modellers, i.e. in order to have solid bodies each edge must be adjacent to two faces and the surface must be closed. Thus, the implemented C# algorithm firstly generates the maximal partitions over the boundary of each component and then merges partitions having the same semantic parameters (e.g. adjacent cylinders having same axis and radius, adjacent planes having the same position and normal etc.) (Figure 7).

Through the import operation (see the data flow presented in Figure 8 and detailed in [Iacob, 2012] for the ViPAD platform) the 3D model of the assembly is analysed and all the geometrical information is combined in a set of lists. Thus, the *Import* module provides input data to the *Interfaces* module. At this stage, an automated identification of the contacts between parts is performed. Using an extended algorithm, the information related to each contact: geometric constraints, contact surfaces relative position, common area etc. is combined in a complete set of interfaces for the assembly, being further used by the *Mobility* module.

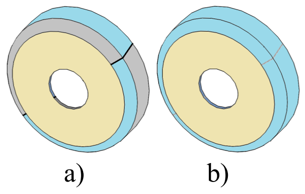


Figure 7. Maximal partitions generation:

a) initial configuration; b) final configuration.

The *Mobility* module implements the mobility operator concept described in the third Chapter and it will be detailed in the following section. This module provides information about the mobility of the components or of the entire assembly.

In parallel, a separated algorithm, using a set of rules, allows the identification of functional parts (such screws, nuts, washers, etc.). The ability to identify functional parts is helpful for reducing the computational times when generating disassembly sequencing, by early eliminating the unfeasible movements.

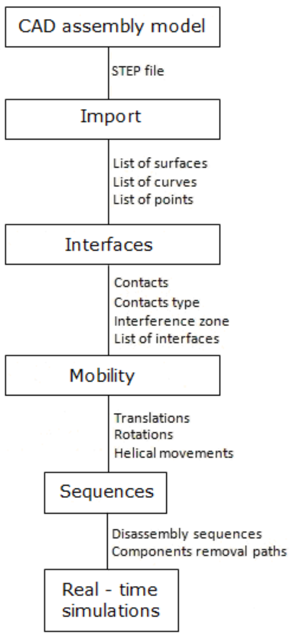


Figure 8. ViPAD software architecture.

Using the previsously computed information, the disassembly sequences in the *Sequences* module are generated considering that the removal of a component from an assembly is made by first breaking the connection interface between target component and neighbours, and secondly, by moving away the component from the assembly along a direction or combination of collision free directions.

All this data can be further used for the real-time simulation of the A/D operations in an immersive environment – the *Real-time simulations* module, which is not yet implemented in the software.

The proposed software tool takes advantage of the component mobility information and represents a new and fast analysis method of the mechanical products disassemblability.

4.2 Mobility module

The developed mobility operator is used to describe and represent all the valid displacements – translations, rotations and helical movements, for a component or for the entire product. It is deployed through the unit sphere and unit ball concepts, for geometrical and graphical representations. Rotations and translations are depicted using two spheres – spherical valid domains, while the helical movements are geometrically illustrated using a volumetric valid domain. Thus, it is possible to capture the nature of the joints in a mechanism and combine their mobilities in order to characterize the full set of trajectories for inserting or extracting a component. Also, a representation of the family of trajectories for a given interface has been developed with a geometric domain describing the set of the disassembly directions.

The pseudo-code implemented in the software application for calculating the mobility of a target component is presented below. It uses the information from the *Interfaces* module and it is based on the methods detailed in the Chapter 3.

```
SR: select component to check mobility
  Selection of CompToCheck
SR: create list of bodies to check
  for each Component in List of components
    if (intersection (CompToCheck, Component) □ 0)
          then Add Component to List of ComponentsIntersection
SR: search for elementary mobilities
  for each Element in List of ComponentsIntersection
    search for rotations
    if (rotations parameters (CompToCheck, Element))
          then Add families to List of Rotations
    search for translations
    if (translations parameters (CompToCheck, Element))
          then Add families to List of Translations
    search for helical
    if (helical parameters (CompToCheck, Element))
          then Add families to List of Helical
SR: compute mobility
  Compute mobility (Rotations, Translations, Helical)
```

The method of generating disassembly directions set is a constructive one. Thus, the geometrical data for each interface of a component is converted into its mathematical representation and then it is sequentially combined with the corresponding data of the adjacent contacts. The result of this series of operations is the component mobility. A similar approach is used for computing the mobility of the entire assembly (product).

4.3 Results / case study

The utility of the developed software tool is demonstrated using a simple, yet general example – the clamping assembly of a robot vacuum gripper (Figure 9 – Photo courtesy of J. Schmalz GmbH).

Figure 9. Schmalz robot vacuum gripper.

Figure 10 shows the display of the list of contact interfaces automatically generated by the software based on the imported 3D CAD assembly model. Thus, the clamping assembly contains 26 interfaces: 14 Planar joints and 12 Helical joints.

For a selected component, the software calculates its mobility relative to neighbours and graphically displays the removal direction/s. For example, as presented in Figure 11, the selected component named "Flansa Schmalz FxC - surub M8x16_hex<3>" has only one direction of mobility – a helical movement, represented on the Helical Movements ball. Further, in order to extract the component from the assembly, the path planner uses the removal direction/s provided by the mobility operator, determines if this is a collision free trajectory (i.e. if the component when moved along the direction/s is not colliding with other components), and then displays the trajectory/ies.

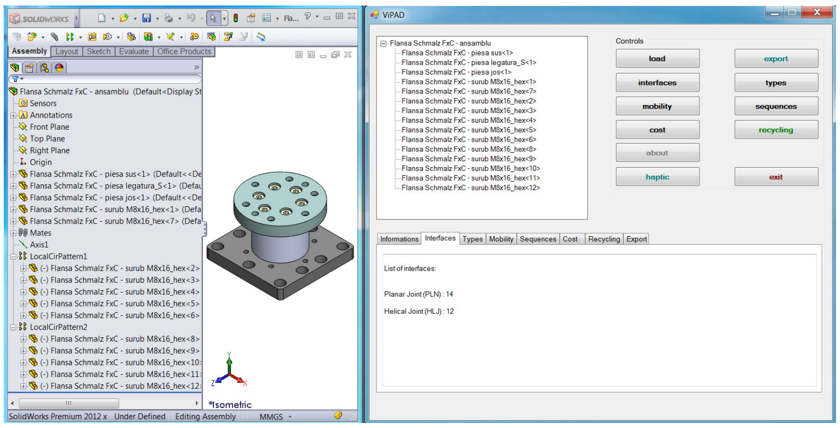
Information on functional parts and on dismounting trajectory for the assembly components are used by the *Sequences* module which implements the disassembly sequence generator for the whole assembly.

5. Conclusions and Further Work

The paper presents the concept of mobility operator and its use in the evaluation of components disassembly feasible directions. The novelty of our approach is related to the fact that this mobility operator can represent all the general movements: translations, rotations, helical movements, required for performing dismounting operations of mechanical assemblies. The mathematical description method and the geometrical representation of the mobility operator have been detailed and a combination algorithm has been proposed.

Data on geometric constraints, contact surfaces relative position and common area between components (gathered in a complete set of interfaces for an assembly), along with information provided by the mobility operator supporting the user to identify components removal trajectories, are all integrated in a software tool.

Further work will be focused on the complete development of the *Real-time simulations* module that will incorporate haptic devices,



Flansa Schmalz FxC - ansamblu (Default<Display St
Sensors
Annotations
Front Plane
Top Plane
Right Plane
Origin
Flansa Schmalz FxC - piesa sus<1> (Default<<De
Flansa Schmalz FxC - piesa legatura_S<1> (Defau
Flansa Schmalz FxC - piesa jos<1> (Default<<De
Flansa Schmalz FxC - surub M8x16_hex<1> (Defa
Flansa Schmalz FxC - surub M8x16_hex<7> (Defa
Mates
Axis1
LocalCirPattern1
(-) Flansa Schmalz FxC - surub M8x16_hex<2>
(-) Flansa Schmalz FxC - surub M8x16_hex<3>
(-) Flansa Schmalz FxC - surub M8x16_hex<4>
(-) Flansa Schmalz FxC - surub M8x16_hex<5>
(-) Flansa Schmalz FxC - surub M8x16_hex<6>
LocalCirPattern2
(-) Flansa Schmalz FxC - surub M8x16_hex<8>
(-) Flansa Schmalz FxC - surub M8x16_hex<9>
(-) Flansa Schmalz FxC - surub M8x16_hex<10
(-) Flansa Schmalz FxC - surub M8x16_hex<11
(-) Flansa Schmalz FxC - surub M8x16_hex<12

SolidWorks Premium 2012 x Under Defined Editing Assembly MMGS

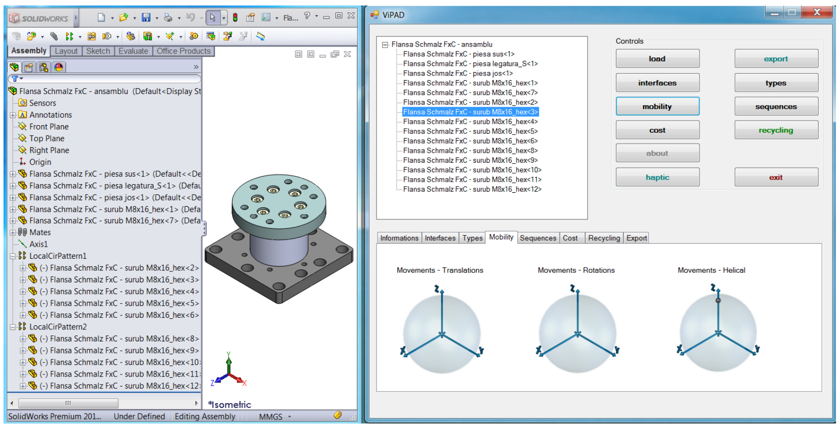
Flansa Schmalz FxC - ansamblu
Flansa Schmalz FxC - piesa sus<1>
Flansa Schmalz FxC - piesa legatura_S<1>
Flansa Schmalz FxC - piesa jos<1>
Flansa Schmalz FxC - surub M8x16_hex<1>
Flansa Schmalz FxC - surub M8x16_hex<7>
Flansa Schmalz FxC - surub M8x16_hex<2>
Flansa Schmalz FxC - surub M8x16_hex<3>
Flansa Schmalz FxC - surub M8x16_hex<4>
Flansa Schmalz FxC - surub M8x16_hex<5>
Flansa Schmalz FxC - surub M8x16_hex<6>
Flansa Schmalz FxC - surub M8x16_hex<8>
Flansa Schmalz FxC - surub M8x16_hex<9>
Flansa Schmalz FxC - surub M8x16_hex<10>
Flansa Schmalz FxC - surub M8x16_hex<11>
Flansa Schmalz FxC - surub M8x16_hex<12>

List of interfaces:

Planar Joint (PLN) : 14

Helical Joint (HLJ) : 12

Figure 10. Identification of contact interfaces.



Flansa Schmalz FxC - ansamblu
Flansa Schmalz FxC - piesa sus<1>
Flansa Schmalz FxC - piesa legatura_S<1>
Flansa Schmalz FxC - piesa jos<1>
Flansa Schmalz FxC - surub M8x16_hex<1>
Flansa Schmalz FxC - surub M8x16_hex<7>
Flansa Schmalz FxC - surub M8x16_hex<2>
Flansa Schmalz FxC - surub M8x16_hex<3>
Flansa Schmalz FxC - surub M8x16_hex<4>
Flansa Schmalz FxC - surub M8x16_hex<5>
Flansa Schmalz FxC - surub M8x16_hex<6>
Flansa Schmalz FxC - surub M8x16_hex<8>
Flansa Schmalz FxC - surub M8x16_hex<9>
Flansa Schmalz FxC - surub M8x16_hex<10>
Flansa Schmalz FxC - surub M8x16_hex<11>
Flansa Schmalz FxC - surub M8x16_hex<12>

SolidWorks Premium 201... Under Defined Editing Assembly MMGS

Figure 11. Graphical representation of component removal direction/s based on mobility operator.

on optimizing the integration of all the modules and on developing, based on the same approach and concepts, a solution CAD platform independent.

Acknowledgements

This work was supported by CNCSIS-UEFISCSU, project number PN-II-RU-233-2010, project title: "Assembly/Disassembly Process Modeling", project type: "Research projects for stimulation of the founding/forming of young independent research teams".

REFERENCES

1. AGUINAGA, I., D. BORRO, L. MATEY, **Path-planning Techniques for the Simulation of Disassembly**, Assembly Automation vol. 27(3), 2007, pp. 207-214.

2. CHUNG, C., Q. PENG, **A Hybrid Approach to Selective Disassembly Sequence Planning for de-Manufacturing and its Implementation on the Internet**, International Journal of Advanced Manufacturing Technology vol. 30(5-6), 2006, pp. 521-529.

3. CHASLES, M. F., **Note sur les propriétés générales du système de deux corps semblables entr'eux et placés d'une manière quelconque dans l'espace; et sur le déplacement fini ou infiniment petit d'un corps solide libre**, Bulletin des Sciences Mathématiques, Astronomiques Physiques et Chimiques vol. 14, 1830, pp. 321-326.

4. DONG, T., L. ZHANG, R. TONG, J. DONG, **A Hierarchical Approach to Disassembly Sequence Planning for Mechanical Products**, International Journal of Advanced Manufacturing Technology vol. 30, iss. 5-6, 2003, pp. 507-520.

5. DUTA, L., F. G. FILIP, **Control and Decision-making Process in Disassembling used Electronic Products**, Studies, Informatics and Control Journal vol. 17(1), 2008, pp. 17-26.

6. GUNGOR, A., S. GUPTA, **Issues in Environmentally Conscious**

Manufacturing and Product Recovery: a Survey. Computers & Industrial Engineering vol. 36, 1999, pp. 811-853.

7. IACOB, R., P. MITROUCHEV PETER, J. C. LEON, **Assembly Simulation Incorporating Component Mobility Modelling based on Functional Surfaces**, International Journal on Interactive Design and Manufacturing vol. 5, 2011, pp. 119-132
8. IACOB, R., D. POPESCU, P. MITROUCHEV, **Assembly/Disassembly Analysis and Modelling Techniques: a Review**, SV Journal of Mechanical Engineering vol. 58(11), 2012, pp. 653-664.
9. ILGIN, M. A., S. GUPTA, **Environmentally Conscious Manufacturing and Product Recovery (ECMPRO): a Review of the State of the Art**, Journal of Environmental Management vol. 91, 2010, pp. 563-591.
10. JIMENEZ, P., **Survey on Assembly Sequencing: A Combinatorial and Geometrical Perspective**, Journal of Intelligent Manufacturing vol. 24(2), 2013, pp. 235-250.
11. LAMBERT, A. D., **Disassembly Sequencing: A Survey**, International Journal of Production Research vol. 41(16), 2003, pp. 3721-3759.
12. LI, J. R., L. P. KHOO, S. B. TOR, **An Object-oriented Intelligent Disassembly Sequence Planner for Maintenance**, Computers in Industry vol. 56(7), 2005, pp. 699-718.
13. LI, J. R., L. P. KHOO, S. B. TOR, **Generation of Possible Multiple Components Disassembly Sequence for Maintenance using a Disassembly Constraint Graph**, International Journal of Production Economics vol. 102(1), 2006, pp. 51-65.
14. MASCLE, C., H. P. ZHAO, **Integrating Environmental Consciousness in Product Development based on Life-Cycle Thinking**, International Journal of Production Economics, vol. 112, 2008, pp. 5-17.
15. MO, J., Q. ZHANG, R. GADH, **Virtual Disassembly**, International Journal of CAD/CAM vol. 2(1), 2002, pp. 29-37.
16. PERRET, J., C. KNESCHKE, J. VANCE, G. DUMONT, **Interactive Assembly Simulation with Haptic Feedback**, Assembly Automation vol. 33(3), 2013, pp. 214-220.
17. POMARES, J., S. T. PUENTE, F. TORRES, F. A. CANDELAS, P. GIL, **Virtual Disassembly of Products based on Geometric Models**, Computers in Industry vol. 55(1), 2004, pp. 1-14.
18. PUENTE, S. T., F. TORRES, O. REINOSO, L. PAYA, **Disassembly Planning Strategies for Automatic Material Removal**, International Journal of Advanced Manufacturing Technology vol. 46(1), 2010, pp. 339-350.
19. SANTOCHI, M., G. DINI, F. FAILLI, **Computer aided Disassembly Planning: State of the Art and Perspectives**. CIRP Annals – Manufacturing Technology vol. 51, 2002, pp. 507-529.
20. SETH, A., J. VANCE, J. H. OLIVER, **Virtual Reality for Assembly Methods Prototyping: a Review**. Virtual Reality, vol. 15(1), 2011, pp. 5-20.
21. SMITH, S. S., W. H. CHEN, **Rule-based Recursive Selective Disassembly Sequence Planning for Green Design**, Advanced Engineering Informatics vol. 25(1), 2011, pp. 77-87.
22. TAKEUCHI, S., K. SAITOU, **Design for Product Embedded Disassembly**, Studies in Computational Intelligence vol. 88, 2008, pp. 9-39.