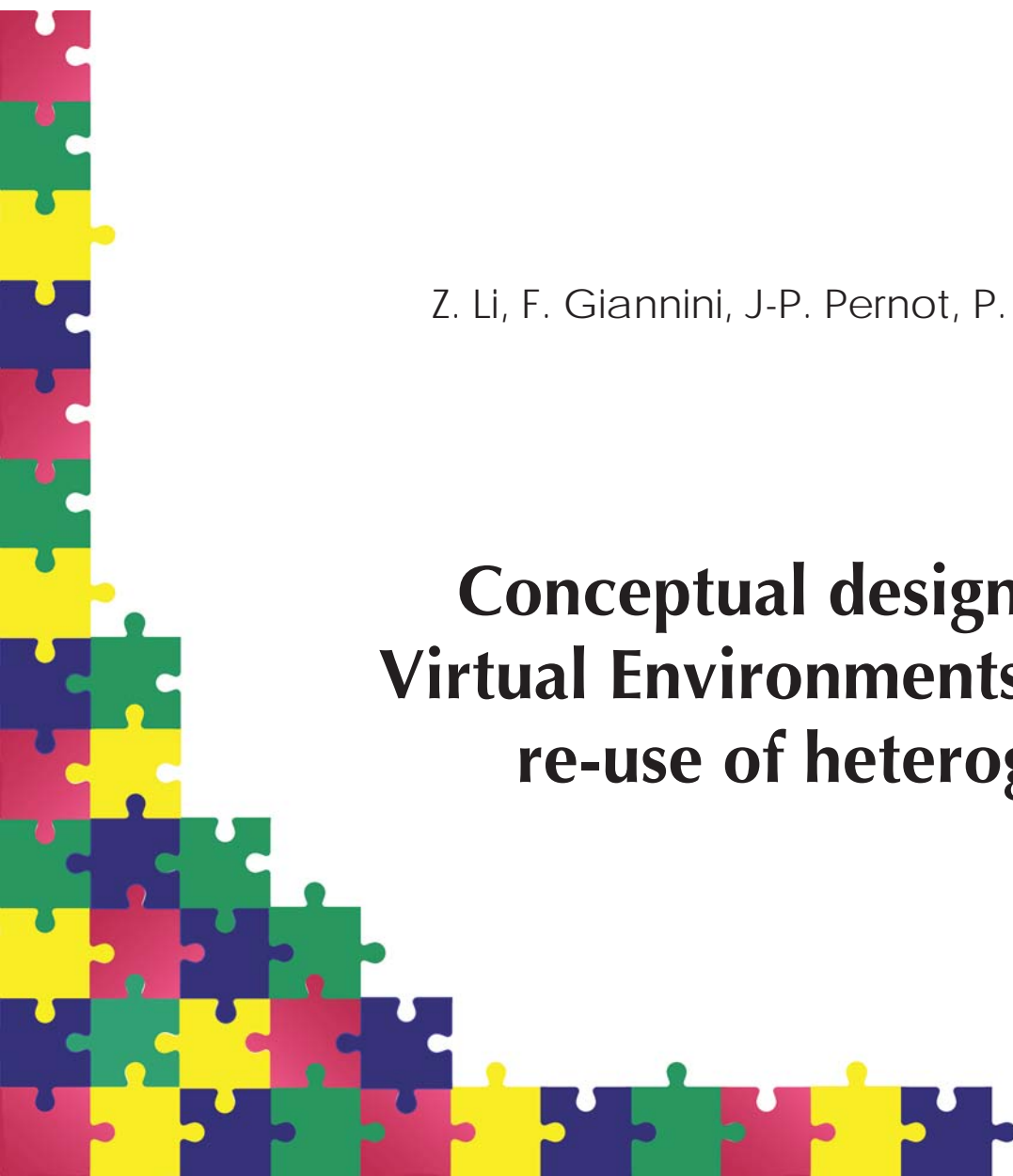


## REPORT SERIES

Z. Li, F. Giannini, J-P. Pernot, P. Véron, B. Falcidieno

# Conceptual design of shapes in Virtual Environments through the re-use of heterogeneous data



# IMATI REPORT Series

Nr. 16-17 – November 2016

## **Managing Editor**

Paola Pietra

## **Editorial Office**

Istituto di Matematica Applicata e Tecnologie Informatiche “E. Magenes”

Consiglio Nazionale delle Ricerche

Via Ferrata, 5/a

27100 PAVIA (Italy)

Email: [reports@imati.cnr.it](mailto:reports@imati.cnr.it)

<http://www.imati.cnr.it>

Follow this and additional works at: <http://www.imati.cnr.it/reports>

---

Copyright © CNR-IMATI, 2016.

IMATI-CNR publishes this report under the Creative Commons Attributions 4.0 license.

Conceptual design of shapes in Virtual Environments  
through the re-use of heterogeneous data

*Z. Li, F. Giannini, J-P. Pernot, P. Véron, B. Falcidieno*



Zongcheng Li:

[Zongcheng.LI@ensam.eu](mailto:Zongcheng.LI@ensam.eu)

Jean-Philippe Pernot:

[jean-philippe.pernot@ensam.eu](mailto:jean-philippe.pernot@ensam.eu)

Philippe Véron:

[philippe.veron@ensam.eu](mailto:philippe.veron@ensam.eu)

Bianca Falcidieno:

[bianca.falcidieno@ge.imati.cnr.it](mailto:bianca.falcidieno@ge.imati.cnr.it)

Franca Giannini:

[franca.giannini@ge.imati.cnr.it](mailto:franca.giannini@ge.imati.cnr.it)

Arts et Métiers ParisTech, LSIS UMR CNRS 7296,  
Aix-en-Provence, France

IMATI-CNR,  
Genova, Italy

---

**Abstract.**

Today, digital data such as 2D images, 3D meshes and 3D point clouds are widely used to design Virtual Environments (VE). Most of the time, only one type of those multimodal data is used to describe and specify the shapes of the objects. However, a single object can be seen as a combination of components linked with constraints specifying the relationships and the rigid transformations defining their arrangement. Thus, the definition of new methods able to combine any kind of multimodal data in an easy way would allow non-experts of VE to rapidly mock-up objects and scenes. In this paper, we propose a new shape description model together with its associated constraints toolbox enabling the description of complex shapes from multimodal data. Not only rigid transformations are considered but also scale modifications according to the specified context of the constraint setting. The heterogeneous virtual objects (i.e. composed by scalable multimodal components) then result from the resolution of a constraint satisfaction problem through an optimization approach. The proposed approach is illustrated with examples obtained with our prototype software.

**Keywords:** *Virtual Reality, conceptual design, shape and object description, heterogeneous data*

---

*[page left intentionally blank]*

## 1. Introduction

Due to the great advances in acquisition devices and modeling tools, a huge amount of digital data (e.g. images, videos, 3D models) is becoming now available in various application domains. In particular, Virtual Environments (VE) make use of those digital data allowing more attractive and more effectual communication and simulation of real or not (yet) existing environments and objects. Despite those innovations, the design of application-oriented virtual environment still results from a long and tedious iterative modeling and modification process that involves several actors (e.g. experts of the domain, 3D modelers and VR programmers, designers or communications/marketing experts). Depending on the targeted application, the number and the profiles of the involved actors may change. Today's limitations and difficulties are mainly due to the fact there exists no strong relationships between the expert of the domain with creative ideas, the digitally skilled actors, the tools and the shape models taking part to the VE development process. Actually, existing tools mainly focus on the detailed geometric definition of the shapes and are not suitable to effectively support creativity and innovation, which are considered as key elements for successful products and applications. In addition, the huge amount of available digital data is not fully exploited. Clearly, those data could be used as a source of inspiration for new solutions, being innovative ideas frequently coming from the (unforeseen) combination of existing elements. Therefore, the availability of software tools allowing the re-use and combination of such digital data would be an effective support for the conceptual design phase of both single shapes and VR environments.

Idea generation techniques can be perceived as the first and most critical part of creative design. Most of the time, the innovative ideas are generated by iterating back and forth between multiple sources. Smith [1] has summarized 172 methods for generating ideas. Actually, the most creative ideas are coming from copying and combining existing things [2] as Albert Einstein always said about his thoughts: *"Words do not play any role in my thought; instead, I think in signs and images which I can copy and combine."* (Albert Einstein).

This idea has been used in shape modelling. Jain et al. have set up a system to create new shapes by blending between shapes taken from a database [3]. Similar approaches can also be found in sketch-based modeling and search system such as presented in [4] and [5]. Therefore, taking ideas from different compositions, then combining and rearranging them together with specific relations and structures is a very common and popular way in creative conceptual design. Nevertheless, such a combination problem becomes difficult when combining heterogeneous data as it will be discussed in the next section. This paper addresses such a difficult problem. A new approach and system have been proposed and validated to enable the conceptual design of VE and associated digital assets by combining existing shape resources while keeping their associated semantics.

The paper is organized as it follows. Section 2 reviews the state of the art in this domain. The proposed Generic Shape Description Model is then introduced in section 3 together with its constitutive elements. This new modelling approach is then illustrated through several examples mixing different heterogeneous data according to different scenarios. The last section concludes this paper and gives directions for future works.

## 2. Related works

### 2.1 Towards modeling with heterogeneous data

Even if there exist approaches to model by composition, those methods do not really refer to combining heterogeneous data in the sense that we intend to do, i.e. with each specific type of input considered as a part of the shape. Actually, heterogeneous data are used in different ways in today's modeling approaches.

With the development of 3D scanners and 3D printers, the techniques related to point cloud meshing are highly required. Point clouds, a kind of heterogeneous data, are considered as input, and then different meshing approaches are applied to generate a mesh. However, generally, in few applications that point clouds are mixed with other data such as meshes.

Another path to modeling using different types of data is image-based modeling (IBM), which reconstructs a 3D mesh from 2D images. Due to the fact that IBM consists a good potential for generating very realistic images, it has gained a lot of attention in the computer graphics community. In particular, IBM techniques are usually used on constrained problem of reconstructing architectural models ( [6] [7] [8] [9]).

One situation in which both 2D and 3D shapes are used together in a same model consists in taking a 2D image as texture for 3D models. The related techniques are named as texture mapping, which is a way of adding surface details, texture (a bitmap or raster image), or color to computer generated graphics or 3D models. Texture mapping has made it possible to simulate near-photorealistic 3D models in real time. This is one of the very common way to represent 2D shapes and 3D shapes all together. In most 3D video games, 2D planar surfaces with transparent texture mapping are usually used together with closed 3D surfaces to simulate objects (e.g. grass, leaves or trees ).

For CAD (Computer-Aided Design) usage, 2D planar surfaces are also used to present different views of the designed product. These 2D planar surfaces are positioned freely without defining any specific relations between them. They are not usually used to design a shape but simulate it in an approximate way.

Text as a kind of shape is also used together with 2D or 3D shape in VR applications. For 2D shapes, Microsoft Word and Microsoft PowerPoint are very common examples which enable to manipulate texts with 2D shapes. However, these texts are not used to define a shape but to present information in addition to the underlying context.

In 3D multiplayer games texts are usually put above a character as its name or conversations. They can also be considered as a planar surface that always faces the viewer. However, texts are not really used today to design new shapes but their semantic meaning are somehow treated as descriptive sentences to describe new shapes [10].

## **2.2 Structure-based shape descriptors**

Most shape description techniques consider information related to the contours or the regions of the shape. Some of these techniques may transform 2D/3D space coordinates into other space to get useful information. Color and light information are also used to describe a shape. Additionally algorithms have been developed providing structure-based and more meaningful descriptors. As those techniques are typically used for extracting information from a well-defined shapes, their main applications are shape classification and retrieval. However, among these techniques, there are some whose results can also be potentially used for modeling new shapes especially graph- or structure- based techniques, which might turn out to be very useful to align or assemble different shapes [11].

Shape skeleton is a thin version of a shape, obtained from points which hold the same distance to its boundaries. There are several mathematic definitions used in the literature to define a skeleton. Different algorithms have been applied to compute the skeleton. The concept of skeleton is also interchangeable as “medial axis” and “thinning”. Reeb graphs [12] [13] represent the evolution of the level sets of a real-valued function defined over the surface bounding the object. Reeb graph, as it strongly preserves the topological information of a shape, has been widely used in different areas. If the function used to calculate reeb graph is on a special flat space, then the results forms a polytree which is also named as a contour tree. As skeletons, Reeb graph is also helpful for image segmentation.



Those graph-based shape descriptors have a strong potential usage to define or align shapes. For example, the one straight segment of a skeleton may represent the major orientation axe of this shape. If the shape is used and relocated in another 3D space, then its skeleton is very useful to set the orientation of this shape.

### 2.3 A multi-layered shape understanding paradigm

Shape representations and description techniques have shown different ways of capturing information from shapes with different aspects. Those features can also be considered as different characteristics for understanding the information associated with shapes. With the development of computer graphics and its heightened application domains, the meaning of “Shape” has become richer.

In [14], shape is defined by “Parts” and “Relations”. The shape is seen as a set of parts that are spatially arranged through the spatial relations among them. The spatial relations among the shape parts can be classified in different ways [15] [16]. A possible classification proposed by [17] includes the following three types of relation: topological, distance and directional. Topological relation, such as “inside”, “outside” and “adjacent”, is invariant to rotation and scaling transformation. Distance relation is linked to quantitative measures. The meaning that two shapes being “far from” or “close to” each other needs to be further specified. Directional relation is characterized by the orientation of angle-based aspects following some reference such as the medial axis, or the segments of the border of a 2D shape.

In 2004, the European project AIM@SHAPE [18] proposed a new way of understanding shapes. Shape is any individual object having a visual appearance which exists in some (two-, three- or higher- dimensional) space such as pictures, sketches, images, 3D objects, videos, 4D animations, etc. Shapes are characterized by several properties. Shapes have a **geometry** (the spatial extent of the object), they can be described by **structures** (object features and part-whole decomposition), they have **semantics** (meaning, purpose), and they may also have some **interaction with time** (e.g. history, shape morphing, animation, video). They have **attributes** (colors, textures, names, attached to an object, its parts and/or its features).

Compared with the definition of [14], this definition shows a broader view of shape. The shape parts and their relations can be considered as the structure of shape. Their appearance features such as their colors, textures etc. are grouped as the attributes of the shape. This definition also associates semantics to shapes, which can be used for semantic- based retrieval processes. With this definition, the information associated with shapes can be structured into three different layers including geometric information level, structural information and semantic information levels (Figure 1).

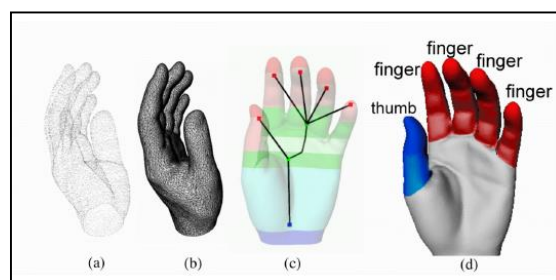


Figure 1. A digital shape represented by two different geometric descriptions: a point cloud (a) and a triangular mesh (b); the structure of the hand model, defined as the configuration of main body with protrusion-like features (c); the corresponding semantically annotated model exploiting its structure (AIM@SHAPE, 2004).

The shape description model proposed in this paper has been designed based on those three layers. This enables the possibility to describe multimodal data in a same structure, so as to be able to combine together all the data whatever their representations are.

### 3. Generic Shape Description Model (GSDM)

The development of conceptual design tools able to specify shapes by composing heterogeneous shape parts requires the specification of a suitable shape representation. To this aim, the so-called Generic Shape Description Model (GSDM) has been defined.

#### 3.1 Overview

The GSDM is structured in three levels of information: **Conceptual Level**, **Intermediate Level** and **Data Level**.

At the Conceptual Level, three basic elements are defined to describe the meaningful object constituents and their relations: **Component**, **Group**, and **Relation**. Component and Group are designed to explain what the different parts of an object are and what the relations between them are. Group indicates shared behaviour or meaning among parts while Relation explains the topological, distance and directional relations of parts. They help the non-expert user to provide an overview of what is going to be described, without requiring a precise specification.

To have a detailed description of each part, the Data Level is needed. This level describes a part of an object through three aspects: **Geometry**, **Structure** and **Semantics**. This information provides the appearance (Geometry and structure) and meaning (Semantics) of a part. This level is only partially handled by the non-expert user and tackles the heterogeneous inputs.

To precise the relation between each part, the Intermediate Level is introduced. At this level, the specific geometric and structural information of a part are addressed. For example, two parts are connected by indicating one's location related to the other. At this level, the whole geometry or the whole structure of each part should be necessarily accessible to the user. To set up those relations, some **Key Entities** should be specified to identify the anchorage elements where restrictions on the related locations are defined. Limitation on reciprocal location between Key Entities are indicated as **Constraint**. All those information are gathered together at the **Intermediate Level**. The end-user does not access to it in a direct manner.

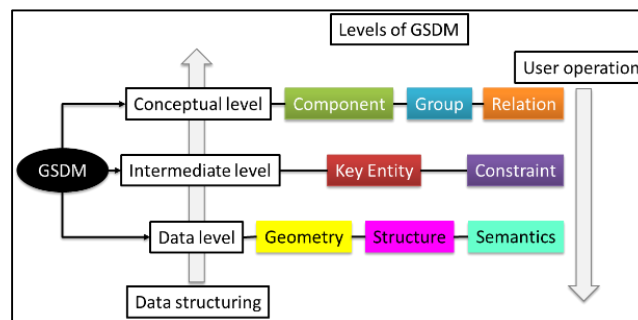


Figure 2. The information levels of GSDM

The different levels and definitions are represented on Figure 2. From the user's point of view, the Conceptual Level is the most convenient level where to work. The user can also work at the Intermediate Level to detail the object's sub-parts arrangement. The Data Level is used for representing the heterogeneous information, for specifying Key Entities, for visualizing Components and for modifying the shape of Components if needed.

Constraints and Key Entities are based on the Data Level and have their own structures. Components, Groups and Relations are more complex in the GSDM definition since they are expressed in terms of the lower level information.

To summarize, in the GSDM, three levels are defined and gather together some elements. All these elements work together to describe a conceptual model of an object that is very convenient for the end-user. The following sections illustrate the different elements of the GSDM as well as the relations between them.

### **3.2 Data Level (Geometry, Structure and Semantics)**

#### *3.2.1 Geometry*

Geometry is the spatial extent of an object represented by different geometric shape representations. The Geometry of an object can be represented by different geometric representations. For example, mathematical expression can be used to represent a Geometry, such as a sphere surface ( $x^2+y^2+z^2 = a > 0$ ).

Geometry stored in the GSDM can be heterogeneous. No assumption is done on the type of data that can be used to represent the shape. This means that at this level, vector and raster 2D and 3D data are addressed all together, which is different from existing techniques. Moreover, different geometric representations can be used to describe the same object's Geometry.

#### *3.2.2 Structure*

Structure can be defined as the relationships between the spaces taken by different parts of an object and are represented by structural-based shape descriptors. It is used to help the user to position different parts possibly defined by heterogeneous data.

Different kinds of structural representations, such as the medial axis, the symmetry axis, the Reeb graph, the skeleton, etc. are convenient to help the user specifying the relative positioning of the heterogeneous parts constituting an object. In the traditional CAD modeling, the structure information such as the axis of a cylinder, the center point of a circle, is used to perform the assembly of different parts. In our approach, the structure information can also support the assembly similarly to what exists in CAD modeling. Actually, in one way, the structure information of the CAD model used for the traditional CAD assembly are the elements used to define this CAD model itself. For example, in the case of a sphere surface defined by a center point and a radius, the center point can be used to constrain the assembly. In the other way, our structure is defined from the geometric representations but is independent from them. It can be kept for future detailed design phase.

The GSDM is used for the conceptual design, which is an early phase of the design process of a new object focusing on the innovation and the creativity aspects. Therefore, the GSDM is not a complete 3D model, but has a multi-layered structure where some parts may be just a 2D model. However, those 2D models can be used by reverse engineering approaches to build a real 3D model during the detailed design phases.

Finally, the Structure defined in the GSDM is also heterogeneous. At the Data Level, the Structure of the GSDM can be a combination of different shape descriptors, such as the medial axis or segmentation, obtained from the corresponding geometric data, either 2D or 3D. Examples of different structural representations are shown in Figure 3.

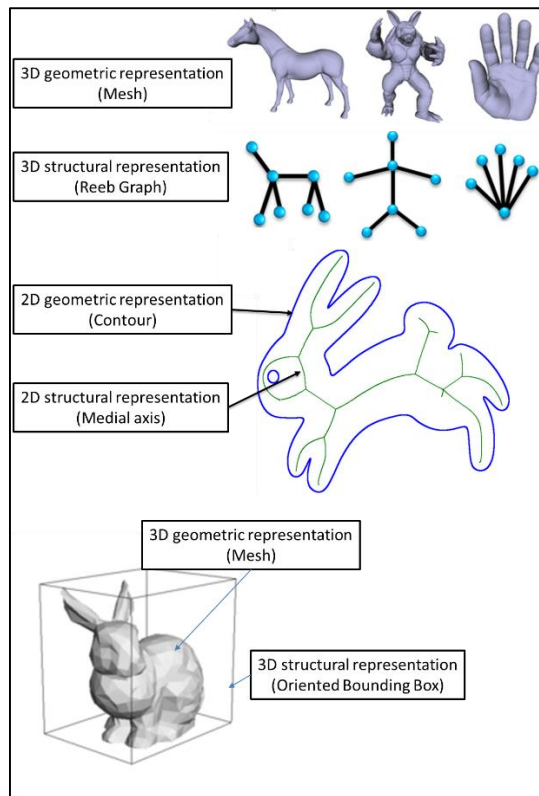


Figure 3. Examples of geometric and structural representations

### 3.2.3 Semantics

Semantics is the purpose and meaning of an instance or an action in a specific context. For example, it can be used to either define the names of the parts used in the conceptual design, or to specify the intention of an instance or of an action. Considering our GSDM, different information are reorganized together following user-specified rules. Most of the time, these rules are associated with meanings explaining why this action is done. For example, the user wants to put two parts together. This action of putting things together can be associated to the purpose of geometrically merging the two parts into one or it can have the purpose of assembling them such that each part maintains its individuality.

Semantics can also be used for further design phase or information retrieval. There are two kinds of semantics: intrinsic and extrinsic. Intrinsic semantics express the meaning of something that can be obtained directly from it. For example, a surface can be considered as cylindrical if the distances between all the points on the surface to an axis are equal. The intrinsic semantics in the GSDM can be the “type” of the geometry or structure. This intrinsic information can be extracted from the original data with more or less complex computations. Extrinsic semantics refers to additional information independent of the original data under a specific context. For example, some additional information such as the color, the material, the name, the function, the role in the overall object (e.g. chair, seat, legs) can be added to the representations depending on the context. Those information are not contained in the instance and have therefore to be attached/added to it.

### 3.3 Conceptual Level (Component, Group and Relation)

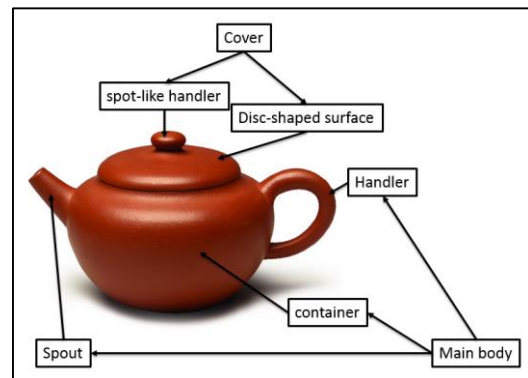
The Data Level presented in the previous subsections contains the basic information of a shape needed for its visualization and manipulation. However, in our system it is not directly operated by the users who are focusing on the overall conceptual specification of the object to be designed and not in fine-tuning its final shape. The user is more focusing on the part-whole decomposition of the object to which behaviors can be directly associated. This motivates the need of a conceptual level manipulated directly by the non-expert user to specify the decompositions into parts and the relationships between them.

### 3.3.1 Component

In the design context, a Component is a constituent of an object, which re-organizes some (possibly one) Geometric and/or Structural representations together so as to represent a same part with a basic Semantic meaning.

Components are indivisible parts. Here, indivisible means that the user will not decompose this part furthermore. The user can define a part according to its functions or any other purpose. A part can also be split into more parts. At one moment, as the decomposition of a part offers enough information or further decomposition is meaningless, the user will not decompose it anymore. At this time, this part can be considered as an inseparable part. For the example presented in

Figure 4, each time an element composing the shape of the object is further specified, at least two new parts are added in the description (such as the spout, the container and the handle detached from the main container). The number of parts shows us both the complexity of the object and how much precise a user wants to be.



Context / description	Parts
A teapot with two parts, the main body and a cover.	Main body, Cover
The main body is composed by a spout, a container and a handle	Spout, Container, Handle, Cover
The cover has a disc-shaped surface and a spot-like handler	Spout, Container, Handler, Disc-shaped surface, Spot-like handle

Figure 4. Examples of decomposition

Thus, Components are also used to organize data with respect to the object functionality or re-usable shape constituents independently of their geometric and structural representations. One objective of using the GSDM is to use heterogeneous data and to let the possibility to represent a part using several geometric representations with different structural descriptions.

However, no matter what kind of geometric or structural representation is used, in the part-whole relationship, it is still considered as the same part. Therefore, a higher-level notion than Geometry and Structure was needed.

Components own specific properties. They are representation independent. The idea of what a specific Component looks like can come from different ideas obtained from different heterogeneous data. In this definition, the number of geometric or structural representations of a Component is not limited. This is to say that a Component can have different geometric or structural representations (Figure 5). Components are also context-oriented. An object can be decomposed differently depending on the context as in the example presented in Figure 4. A further detailed decomposition may have more numbers of Components. Therefore, which parts of a shape should be considered as Components is to be decided by the user under a specific design context. Finally, every Component could be in turn described according the previously defined three layers of information: Geometry, Structure and Semantics.

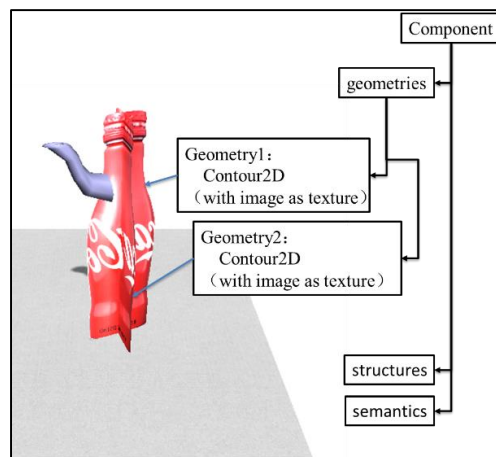


Figure 5. Component with multiple geometric representations

### 3.3.2 Group

Group gathers together several Components associating them a specific meaning or behaviour or attributes. For instance, sometimes, a user wants to treat a set of Components so that they can be selected, modified or searched as a single one. For example, the user may want to change the color of all the Components. In the following we adopt the general term Element to indicate both Components and Groups.

Groups own specific properties. A Group is constituted by at least two Elements, i.e. both Components and Groups can be part of other Groups. All the elements in a Group should have a specified semantic meaning to indicate the purpose of being a Group. It explains why different Elements are to be considered as one. For example, they have the similar function, or they have the same color. A Group can be an Element of another Group and an Element can belong to several Groups. This is the non-exclusive inclusion property. Examples of Groups are presented in Figure 6 which shows a corner of an office room. All the books on the desk can be clustered in a Group called "Books". The laptop and the mouse form a Group called "PC". The "PC" and the "Books" can be also considered as a Group sharing the fact that they all are on the desk. Another Group called "Furniture" refers to all the furniture in this office room including the desk and the chair. The chair and the mouse can be also considered as a Group as they are all made by plastic.

In this example, it can be noticed that "mouse" is shared by four Groups: "PC", "plastic", "on the table" and "Office room". To locate the "mouse" of the Group "Office room", a minimum of one Group ("Office room" -> "plastic" -> "mouse"), and a

maximum of two Groups (“Office room” -> “On the table” -> “PC” -> “mouse”) need to be traversed. The number of Groups to traverse to locate an Element is defined as the depth of the Element in this Group. With different paths, there might be a minimum depth and a maximum depth. Depth characterizes the complexity of a Group and is transparent to the user. However, it could be used by the algorithms for the manipulation of Groups.

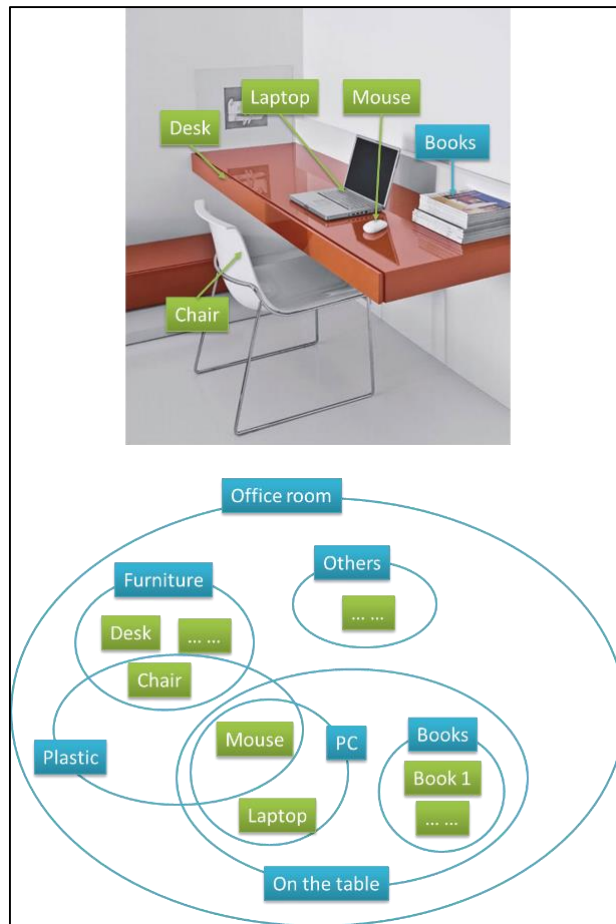


Figure 6. Example of groups [19]. (Blue circles for Groups and green block for Components)

### 3.3.3 Relation

Relations are used to describe the way two Elements (either Components or Groups) are connected together. The links between different Elements may express very complex relations and/or operations, which may require complex and precise information to be achieved. For example, to satisfy the relation expressing the geometric merging of two Components together, the related location of the two Components need to be specified, as well as their geometric representations and the parameters related to the merging algorithm. For a non-expert in Computer Graphics, describing these complex links can be very difficult. One possibility is to describe them top down, i.e. from the purpose to the specification of the Geometric or Structure and associated rules. Additionally, at the conceptual design phase, the purpose of this link is just the indication of the type of relation/operation independently of the representations or of the associated evaluation rules. For the example presented in Figure 7, the user wants to merge the spout and the container of a teapot together as these two Components are connected because water can flow from the container in the spout. Although the two Components have different representations, the link of “merging” exists independently of their underlying representations. At the top level, different types of links indicating the purpose can be specified, and at the low level, the minimal set of details necessary to provide a visual feedback for understanding the wished

outcome need to be stated. Such a minimal set normally includes the related location of the two Components. Relation is focusing on the top level.

The detailed definitions of the considered Relation types are introduced in the following .

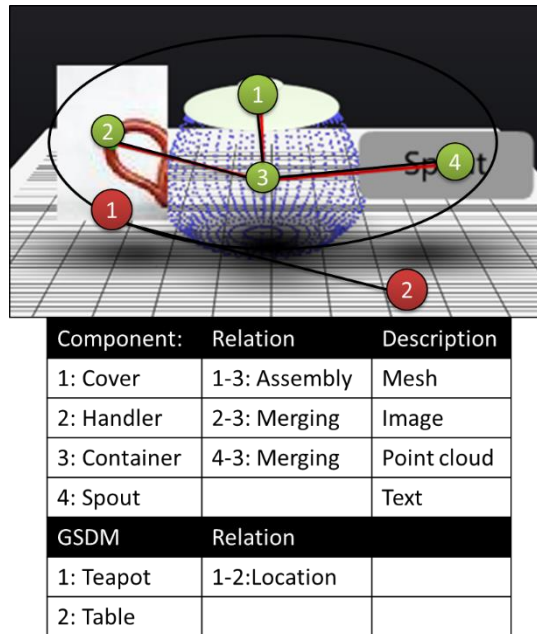


Figure 7. Examples of Relations

The Merging Relation indicates that the two elements have to be geometrically operated to obtain a unique geometric model. It is a very common Relation which corresponds to the Boolean union operation on geometric models. For example, in Figure 7, the teapot is composed by four different Components, each of them having a different geometrical representation. In real life, even if the container and the spout may be produced separately, a merging operation between them can be required such that the spout and the container create a unique continuous volume. In the example of Figure 7, the container is represented by a point cloud, while the spout is represented by a text. At the conceptual design level, this is acceptable since the aim is not to create the final shape of the object but to express all the information needed to fully specify it. This is to allow its complete shape definition at the detailed design phase, without being limited and slowed down by unnecessary modeling details and operations difficult for non-expert users. As already stated, the purpose of the GSDM is to provide the representation of how an object should be created by combining subparts, possibly not completely defined. The relations aim at specifying the links between them. The real Merging operation does not take place at this stage but it can be obtained by processing the GSDM once the geometric description of each constituting Component is completely specified and harmonized (i.e. compatible geometric representations on which Boolean operations can be applied).

The Assembly Relation is a notion similar to what exists in CAD systems. Different Elements are connected together without fusing them into a same Geometry but simply linking them with different joints.

The Shaping Relation is used to indicate the intent to modify the shape of an Element, i.e. to reshape it. It is not simply merging the overlapping area of two Elements by cutting the useless areas as the Merging Relation, but restyling one Element while taking into account the characteristics of another. These two elements may come from different objects. An example is presented on Figure 8 where specific egg-like chairs are obtained while combining a traditional chair with the shape of an egg.



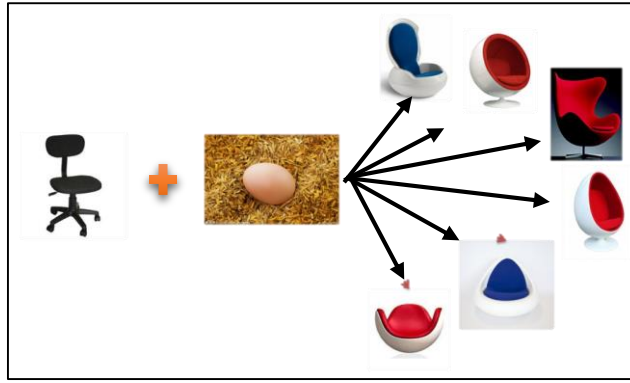


Figure 8. Example of the intent expressed by a relation of “shaping”

The Location Relation is used to position an object with respect to the others. For example, a ball has to lie on the floor. The floor and the ball are not assembled together or merged together. They also keep their own shapes.

Relations own specific properties. A Relation is only built between two Elements. Actually, a Relation can be built between more than two elements by exploiting the Group notion. For example, if four legs of a table need to be assembled to a desktop, a Group “support” can be created including the four legs and it can be linked to the desktop through an Assembly Relation. As already stated, being the Relation aimed at specifying the purpose and rules of the link between Elements, it is independent from the actual representation of each Element. If there is a Relation between Group A and Element B, then this Relation explains that all the Elements in Group A should have the same kind of relation with B or with the Elements in B (if B is a group). This is the inheritance property. For example considering the Group of four legs assembled with a desktop, it is not necessary to indicate that each leg is assembled with the desktop. However, it could be necessary to have some Relations between the legs inside of the Group of legs. This inheritance property is managed at programming level and it is not specified in the data structure. Finally, the uniqueness property indicates that there is only one kind of Relation between two Elements, including the inherited Relation.

### 3.4 Intermediate Level (Key Entity and Constraint)

This level explains how different Components are located in the 3D space the ones with respect to the others. The location of a Component can be represented by the position, orientation and scale of its reference frame (an origin point with three directions). All those values form the unknowns of an optimization problem that is set up using Key Entities and Constraints as explained in the next subsections, as well as a dedicated solving strategy as explained in section 3.5.

#### 3.4.1 Key entity

A Key Entity is a geometric primitive (point, line or an oriented point) associated to either the geometric or structural representations of a Component, or simply located in its local reference frame. Instead of describing the related location of different Components directly between their reference frames, specifying the related location of some featured elements laying on each component offers a more meaningful and natural action of how they are linked.

There exist two categories of Key Entities: Geometric Key Entities and Parametric Key Entities. A Geometric Key Entity of a Component is only related to the local reference frame of the Component, and is not modified when its geometric or structural representation evolves. For example, a Geometric Key Point defined for a mesh corresponds to a position in the local reference frame of the mesh. Thus, if the mesh is scaled, this point will not move. Geometric Key Entities are particularly useful when a Component has no geometric or structural representations. A Parametric Key Entity can be represented by a point, a line or an oriented point so as to represent a plane. These Key Entities can be associated directly to a geometric or structural representation of a Component such as a vertex of a mesh with its normal. However, a Parametric Key Entity can also be created by building

rules between other Key Entities. For example, a line defined by two points, these two points can be Geometric Key Entities or Parametric Key Entities. The new created Key Entity is not directly associated with the geometric or structural representation, thus it is called indirect Parametric Key Entity.

Classification	Point	Line	Oriented Point	Array
Geometric Key Entity (Independent)	$E_P$	$E_L$	$E_F$	\
Parametric Indirect Key Entity	$E_{PP}$	$E_{LP}$	$E_{FP}$	$E_A$
Parametric Direct Key Entity on 2D contour	\	$E_{LC}$	$E_{PoC}, E_{PiC}$	\
Parametric Direct Key Entity on 3D mesh	\	$E_{LM}$	$E_{PM}$	\
Parametric Direct Key Entity	$E_{PW}$	$E_{LW}$	\	\

Table 1. Classification of the proposed Key Entities

Key Entity can be associated not only to a point, a line, an oriented point but also to a combination of them (indicated as array). All the proposed Key Entities are listed in Table 1.

Each Key Entity is defined by some parameters from which the coordinates of the represented geometric primitives are obtained. The following Figure 9 illustrates some examples of the proposed Key Entities.

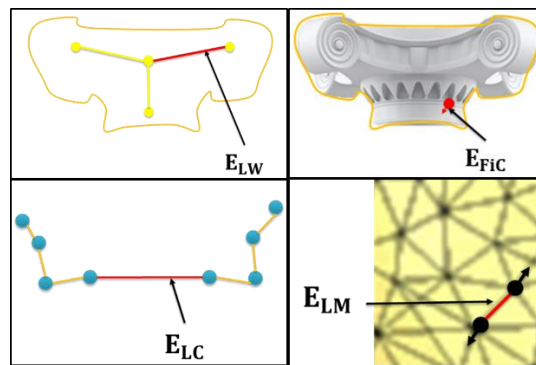


Figure 9 Example of Key Entities

The parameters of a Parametric Direct Key Entity include the related Component, the related representation (geometric, such as 3D mesh, or structural, such as skeleton) and some numerical parameters necessary to compute the coordinates of the associated geometric primitive. The parameters to define a Parametric Indirect Key Entity contain the already specified Key Entities and some numeric parameters explaining their relations.

Key Entities own specific properties. As previously said a Key Entity can be finally represented by a geometric element such as a point, a line, an oriented point or a combination of them (i.e. an array). This indicates the dimension of the Key Entity. A point is a one-dimensional entity, a line and an oriented point are two-dimensional entities, and an array is an n-dimensional entity where n is the sum of its key entities' dimensions. The dimension of a Key Entity can also be represented by the number of  $\mathbb{R}^3$  elements used to specify its representation. For example,  $E_{LC}$  (an edge of a 2D contour) is represented by a line defined by two points

(each of them is an  $\mathbb{R}^3$  space). In other words, each dimension corresponds to an  $\mathbb{R}^3$  instance, which is named as the “dimensional characteristic” of this key entity. All the key entities are represented in a 3D space. A table classifying all the key entities by their dimensions is presented in table 2.

1D Key Entities	2D Key Entities		n-D Key Entities
Point	Line	Oriented Point	Array
$E_P$	$E_L$	$E_F$	$E_A$
$E_{PW}$	$E_{LC}$	$E_{FOC}$	
$E_{PP}$	$E_{LW}$	$E_{FIC}$	
	$E_{LM}$	$E_{FM}$	
	$E_{LP}$	$E_{FP}$	

Table 2. Dimensions of the Key Entities

Finally, from the presented specification of Key Entities, it can be noticed that the Parametric Key Entity can be associated to the geometric and/or the structural representation of a Component, while in traditional CAD systems, the key entities used to specify constraints in assemblies are only located on its geometric layer. This is the multi-modality property.

### 3.4.2 Constraint

Constraints limit the related location of two Key Entities. They are expressed by equations or inequalities linking two Key Entities. If more than two key entities are involved, we use a list of key entities  $E_A$ . To completely specify a Constraint, we need not only to define which are the involved Key Entities, but also which equations or inequalities have to be satisfied. The equations and inequalities depend on the type of constraint. Even for the same Constraint, different combinations of two Key Entities may require different equations or inequalities. In Table 3, all the considered Constraints are shown indicating the related possible combinations of Key Entities.

Symbol	Name	Acceptable Key Entities combination
$C_D$	Distance	(Point, Point), (Point, Oriented point), (Oriented point, Oriented point)
$C_A$	Angle	(Line, Line), (Line, Oriented point), (Oriented point, Oriented point)
$C_{Co}$	Coincidence	(Point, Point), (Point, Oriented point)
$C_{Pa}$	Parallelism	(Line, Line), (Line, Oriented point), (Oriented point, Oriented point)
$C_{Pe}$	Perpendicularity	(Line, Line), (Line, Oriented point), (Oriented point, Oriented point)
$C_{Cl}$	Co-linearity	(Point, Line), (Oriented point, Line)
$C_{Cp}$	Co-planarity	(Point, Oriented point), (Oriented point, Oriented point)

$C_{ca}$	Co-axiality	(Line, Line)
$C_T$	Tangency	( Oriented point, Line), ( Oriented point, Oriented point)
$C_i$	Insertion	(Line, Line)
$C_{ct}$	Contact	(Oriented point, Oriented point)
$C_{Pt}$	Pattern	(Point, Array), (Line, Array), (Oriented point, Array)

Table 3. Constraints and associated Key Entities combination

Coincidence is only between two points. Co-linearity is used to limit the position of a point along a line. Co-planarity is used to limit a point on a surface. Co-axiality limits two lines to be coincident. Insertion is used to put two lines being co-axial then limit the distance between them. Contact is used to put two surfaces touching each other and Tangent is used to limit a line and a plane or two planes to be tangent. Pattern is used to distribute points along a line or around a point.

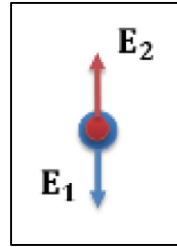


Figure 10. Example of Contact

For example, Contact between two Key Entities E1, E2 is defined as follows (Figure 10):

$$C_{ct}(E1, E2) = e_0(\mathbf{p}_{E1}, \mathbf{p}_{E2}, 1) \ \&\& \ e_3(\mathbf{n}_{E1}, \mathbf{n}_{E2})$$

where  $E_i \in \{E_F, E_{FOC}, E_{FIC}, E_{FM}, E_{FP}\}$  ,  $\mathbf{p}_{E_i}, \mathbf{p}_{E_i}$  are the two points of these two key entities

E1 and E2.  $\mathbf{n}_{E1}, \mathbf{n}_{E2}$  are the normals of these two points

$e_0(\mathbf{V}_1, \mathbf{V}_2, a)$  is a vector equation (3 linear equations)

between two vectors  $\mathbf{V}_1, \mathbf{V}_2$ :  $\mathbf{V}_1 == a \cdot \mathbf{V}_2$

$e_3(\mathbf{V}_1, \mathbf{V}_2)$  three linear equations and three linear

inequalities

between two vectors  $\mathbf{V}_1 = \begin{bmatrix} x1 \\ y1 \\ z1 \end{bmatrix}, \mathbf{V}_2 = \begin{bmatrix} x2 \\ y2 \\ z2 \end{bmatrix}$ :

$$\begin{cases} x1y2 == x2y1 \\ y1z2 == y2z1 \\ z1x2 == z2x1 \\ x1x2 \leq 0 \\ y1y2 \leq 0 \\ z1z2 \leq 0 \end{cases}$$

The constraints that have been considered were thought to be significative for users. As a consequence, semantically, some of them can be special cases of others just putting a specific different value. For example, ‘‘Coincidence’’ between two points can be considered as a special case of ‘‘Distance’’ between two points equals to zero. However, the equation to compute distance is not linear, to maximize the use of linear equations and linear inequalities, ‘‘Coincidence’’ and ‘‘Distance’’ are differently formulated. Therefore, six basic expressions (equations or inequalities) are defined based on vectors, which are used to express all the proposed constraints.

As the other constitutive classes of the GSDM, Constraints own specific properties. As mentioned previously, they are built only between two Key Entities. The value of each Constraint is true or false, in other words it is a Boolean-valued formula. Therefore, conditional operations can be applied between Constraints, such as conditional equal (“==”), conditional AND “&&” and conditional OR (“||”). The results of the conditional operations are still Boolean-valued. Finally, because of the specification of Key Entities, the constraints are built both at the structural and the geometric levels.

### 3.5 Constraint satisfaction solving

The fulfillment of all the Constraints corresponds to solving a Constraint Satisfaction Problem (CSP). During the modeling process with GSDM, user interactions end before CSP solving starts automatically.

Numerical optimization is a way to solve the CSP, which consists of:

- A set of  $n$  variables  $X = \{x_1, \dots, x_n\}$  whose values are to be found;
- For each variable  $x_i$ , a finite set  $D_i$  of possible values (its domain);
- A set of constraints  $C = \{c_1, \dots, c_n\}$  limiting the values that variables can take.

Numerical optimization is used to solve the problem of minimizing or maximizing a function  $f(x)$  subject to constraints  $\Phi(x)$ . Here  $f: \mathbb{R}^n \rightarrow \mathbb{R}$  is called the objective function and  $\Phi(x)$  is a Boolean-valued formula defined on top of the set of constraints  $C$ . Both global and local optimization approaches are used to solve the CSP depending on different application context. Several numerical algorithms can be used to solve the optimization problem. For example, the linear problem, simplex algorithms, revised simplex algorithms, interior point algorithms can be used as presented in [19]. For nonlinear local optimization, the interior point algorithm [20] can also be used. For nonlinear global optimization, Nelder–Mead [21], differential evolution [22], simulated annealing [23] and random search can be used. Therefore, to solve the CSP, first we need to check to which type our CSP belongs.

In our case, as presented previously, Constraints are used to limit the location of all the Components. The location of a Component corresponds to the location of its reference frame in global space which includes its position, orientation and scale factor. Thus, the variables to be determined in our CSP are the position, orientation and scaling of each Component’s local reference frame in the 3D space. Therefore, for each Component there are nine different variables: the 3D positions, 3D orientations and 3D scales. In the proposed approach, the set of variables associated to a Component  $i$  is described mathematically as below :

$$Rf_i = (\mathbf{P}_i, \mathbf{R}_i, \mathbf{S}_i) = \left( \begin{bmatrix} xi \\ yi \\ zi \end{bmatrix}, \begin{bmatrix} \alpha i \\ \beta i \\ \gamma i \end{bmatrix}, \begin{bmatrix} ai \\ bi \\ ci \end{bmatrix} \right)$$

The domain for each variable is different. The positions and scales are real values and the orientations should be from -180 to 180 degrees:

$$D_i = (\mathbb{R}^3, [-\pi, \pi]^3, \mathbb{R}^3)$$

The numerical optimization will give an optimal solution, which needs to minimize an objective function. This choice has been made to be able to access more meaningful solutions compared to traditional CAD systems where only one solution is returned. In real life, energy input is required to relocate an object. For example, to move an object from point A to point B, it is always desirable to spend less energy to realize the desired action. In physics, the energy or work ( $w$ ) spent to move, rotate or deform an object can be described as below.

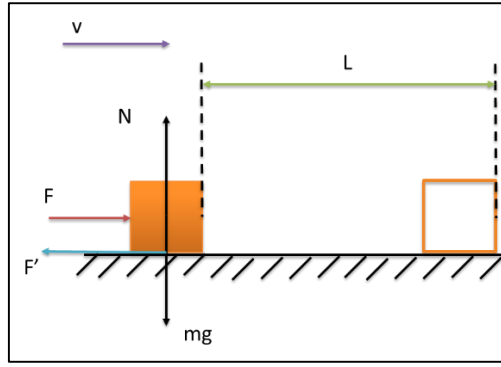


Figure 11. Positioning energy

For the simplest situation as presented in Figure 11, an object is moved on a floor for a distance of “L” along the same direction of a pushing force “F” with a constant speed “v”. Considering “f” is the coefficient of the resistance between the floor and this object, “N” is the supporting force from the floor (where,  $N = mg$ ). “F’ ” is the friction from the floor, the work spent for this positioning can be express as:

$$w_p = F \cdot L = F' \cdot L = f \cdot m \cdot g \cdot L = f \cdot g \cdot \rho \cdot V \cdot L = \mu_p(V) \cdot L$$

$$\mu_p(V) = f \cdot g \cdot \rho \cdot V$$

Where  $g$  is the acceleration of gravity is,  $m$  is the mass of this object,  $\rho$  is the density of the object,  $V$  is the volume of the object. If each Component has the same material and is in the same environment (on a same floor), then  $\mu_p(V)$  can be considered as a factor of positioning energy which is only depending on the volume of this Component.

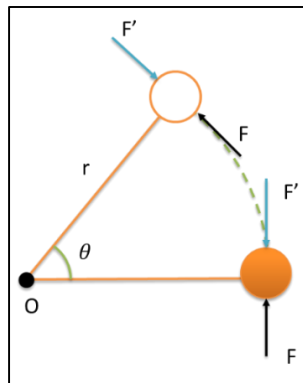


Figure 12. Rotation energy (top view)

Similar to the positioning, one of the most simple situation to rotate an object is presented in Figure 12. It can be described as a rotation of an object around a point “O” with a force “F” and a constant speed. “F’ ” is the friction from the floor, the work used for this rotation can be express as:

$$w_r = F' \cdot r \cdot \theta = f \cdot m \cdot g \cdot r \cdot \theta = f \cdot g \cdot \rho \cdot V \cdot r \cdot \theta = \mu_r(V) \cdot \theta$$

Where,  $r$  is the radius for the rotated arc (the distance between the mass center and the rotation center) and  $\theta$  is the rotated angle,  $g$  is the acceleration of gravity,  $m$  is the mass of this object,  $\rho$  is the density of the object,  $V$  is the volume of the object, “f” is the coefficient of the resistance between the floor and this object. Similarly, if we consider that each Component has the

same material, is in the same environment, and rotates with the same rotating radius, then  $\mu_r(V)$  can be considered as a factor of rotation energy which is only depending on the volume of this Component.

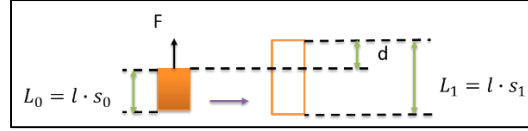


Figure 13. Scaling energy

In our approach, the scale of the Component is also considered as a variable for the CSP solving. Therefore, each component can be considered as a spring (Figure 13). According to Hooke's law, the scaling energy can be expressed as:

$$w_s = \frac{1}{2} \cdot k \cdot d^2 = \frac{1}{2} k (L_1 - L_0)^2 = \frac{1}{2} k (l \cdot s_1 - l \cdot s_0)^2 = \frac{1}{2} k \cdot l^2 \cdot \Delta s^2$$

$$\approx \frac{1}{2} k \cdot S \cdot \Delta s^2 = \mu_s(S) \cdot \Delta s^2$$

Where "F" is the force used to scale the spring, "k" is the coefficient (Young's modules) of the elastic deformation. "d" is the scaled distance along the scaling direction. "l" is the initial length of this object along the direction of "F". "s<sub>0</sub>" and "s<sub>1</sub>" are the scale factors before and after scaling. If we consider  $l^2$  is approximately equals to the section (S) of this scaling  $\mu_s(S)$  can be considered as a factor of scaling energy which is only depending on the section along the scaling direction of this Component.

Inspired by these energies, a global energy has been defined and used to solve the optimization problem. The objective function is thus defined as below:

$$F = W_p + W_r + W_s$$

Where,

$$W_p = \sum_{i=1}^{i=n} \mu p_i \| \mathbf{P}_i^{k+1} - \mathbf{P}_i^k \|$$

$$W_r = \sum_{i=1}^{i=n} \mu r_i \| \mathbf{R}_i^{k+1} - \mathbf{R}_i^k \|$$

$$W_s = \sum_{i=1}^{i=n} \mu s_i \| \mathbf{S}_i^{k+1} - \mathbf{S}_i^k \|^2$$

$n \in \mathbb{N}$  is the number of Components

$W_p$  is the positioning energy of all the Components from the previous position  $\mathbf{P}_i^k$  to the final position  $\mathbf{P}_i^{k+1}$  after constraining. The longer the distance between the previous and the final positions of this Component is, the larger the positioning energy is.  $\mu p_i$  is the positioning energy factor potentially defined for each Component. Inspired from the physical energy of movement presented previously, by default it is equal to the volume of the Component (for an image, instead of using volume, surface is considered), which means if the Component is bigger, then more energy is needed to reposition it.

$W_r$  is the orientation energy of all the Components from the previous orientation  $\mathbf{R}_i^k$  to the final one  $\mathbf{R}_i^{k+1}$ . Similar to the positioning energy, the larger the angle between the two orientations is, the more energy is needed.  $\mu r_i$  is the orientation factor for each Component which also equals the volume of the OBB of the Component (surface area if it is an image), and can also be modified.

$W_s$  is the scaling energy of all the Components from the previous scale  $\mathbf{S}_i^k$  to the final scale  $\mathbf{S}_i^{k+1}$  of the Component.  $\mu r_i$  is the scale factor which is related to the volume of the Component (surface if it is an image).

Our objective is to minimize the sum of these three energies. If the position of a Component  $i$  should not change too much, the  $\mu p_i$  parameter can be set up to a very large value. In this sense, a link between the relocations of each Component and a semantic meaning is set up. In other words, the proposed resolution strategy is more meaningful compared with the one integrated in traditional CAD modelers. The importance of semantics for the constraints could be found in [24]. Thus, different factors can be used for different Components. The energy factors actually limit the flexibilities of positioning, rotating and scaling each Component.

Once the CSP has been specified, different approaches can be applied to obtain the final result.

To summarise, the structure of GSDM can be described as shown in Figure 14.

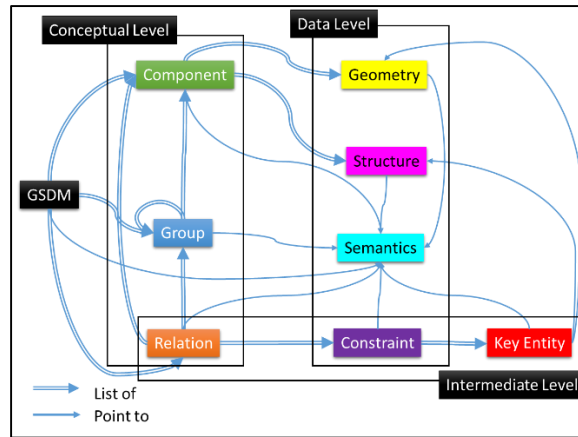


Figure 14 Structure of GSDM

The GSDM has a list of Components, a list of Groups and a list of Relations. Each component includes its geometric and structural representations and its Semantics. Group contains a list of Groups and/or a list of Components together with their Semantics. Relations are structured by a list of Constraints, each of which has two Key Entities pointing to the geometric or structural representations of the related Component.

#### 4. Results

The GSDM introduced in this paper has been implemented on a user-friendly system totally developed by the authors using C# language based on Unity3D [26]. The adopted mathematical tool for solving the CSP is Mathematica 9 [27]. To illustrate the power of the proposed approach, various examples have been tested using this system.

The first example, aiming at illustrating all the representation and functional capabilities, is a crazy chair design with 2D pictures and 3D meshes (Figure 15). From a set of inputs (Figure 15.A) and user-specified Relations linking Key Entities with Constraints, our system generates the solution presented on Figure 15.B. The figures relative to this example are shown in Table 4.



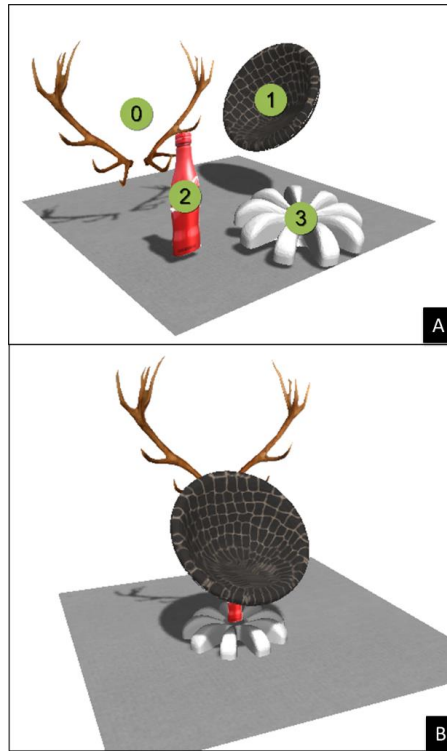


Figure 15 Example one – Crazy chair. A: inputs; B: result of GSDM

The second example (Figure 16) is a mechanical engine, showing the possibility to assemble scanned parts without necessarily reconstructing the CAD models as it is traditionally performed in classical reverse engineering techniques integrated in commercial CAD software.

The third example (Figure 17) is a configuration of a nuclear site, demonstrating the capacity to rearrange 3D models on a 2D plan, which is much useful for architecture design. Here again, in contrary to what is possible in commercial software, our system really solve a set of Constraints specified between Key Entities of the image and Key Entities of the CAD models thus exploiting heterogeneous data.

The following table shows the number of Elements defined in the GSDM of these three examples. It shows that the simultaneous manipulation of heterogeneous data has been made possible and is quite fast with respect to the time the user would have to spend to do it manually.

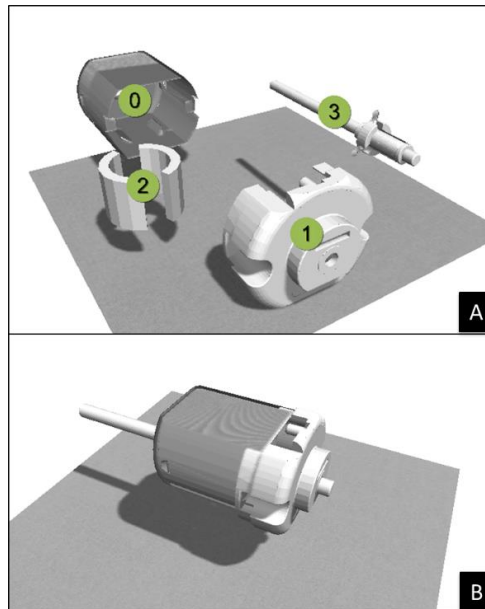


Figure 16 Example two – Mechanical engine. A: inputs; B: result of GSDM

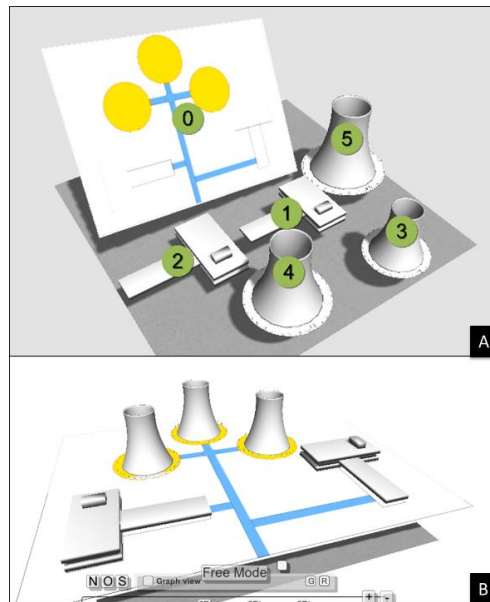


Figure 17 Example three – Nuclear site. A: inputs; B: result of GSDM

	Example 1	Example 2	Example 3
Components	4	4	6
Groups	2	0	0
Relations	3	4	5
Key Entities	11	16	24
Constraints	6	11	20
Linear equations	18	20	40

CSP solving time	12.5s	39.7s	57.9s
------------------	-------	-------	-------

Table 4 Results

## 5. Conclusion and perspectives

This paper has introduced the so-called Generic Shape Description Model (GSDM) together with its general structure and associated concepts and definitions. This is the first step for describing shapes with heterogeneous data using a unified approach. Heterogeneous objects are obtained while constraining different components within a unique reference frame. Position, rotation and scale of the components are considered as unknowns of an optimization problem. An extended constraints toolbox has been developed together with the mechanisms to specify them in an easy way. The resolution of the optimization problem tends to minimize a deformation energy involving position, rotation and scaling factors. The proposed approach has been illustrated through several applications requiring the simultaneous manipulation of heterogeneous models in different context. It is clear that using such an approach is much more efficient and accurate than what exists in commercial CAD software. However, the development of an effective conceptual design tool based on the GSDM requires the resolution of some research and implementation issues.

The semantics associated to the current version is mainly used to store information for initializing different constituents of the GSDM, such as the “type” or “reason”. For some specific design contexts and applications, it can be further specified and extended together with the related mechanisms to treat such high-level information.

The concepts of geometry and structure have been included in the GSDM. In principle, they encompass any geometric and structural representation. In this work, not all the geometric and structural representations have been treated. To effectively exploit all the existing resources, additional representations should be considered. This could be done through the development of new plugins. Moreover, even if there exists plenty of algorithms for shape segmentation and structural descriptors’ computation, most of the data available are still containing only pure geometric information. Actually, most of the resources require some human intervention to be used in our system for the component selection. This is a limitation. Additionally, for input data missing structural information, the system automatically creates a structure that is the bounding box, which might limit the specification of the relations between components

Of course, the constraints toolbox can also be extended to consider new constraints useful in specific applications that have not yet been treated. User-specified constraints can also be considered to extend even more the capacity of the system.

Moreover, the relation type of “Shaping” is not fully expressed in this manuscript, while just a general concept has been proposed. However, such a relation can be of real interest for design and creativity issues. In the post-processing, a fully 3D representation should be generated from the GSDM with its 3D structure and semantics. This requires more advanced techniques in mesh merging and 3D reverse engineering from images. New research on structure and semantics merging is required for their correct updating according to the achieved 3D object model. With both the pre-processing (shape segmentation and structuring) and post-processing phases, we believe that the GSDM can be used in the whole 3D object design process while strongly improving the collaborative conceptual design phase.

Finally, we can imagine to use GSDM in other application contexts, such as the medical analysis domain for representing different medical data and diagnostic results (CT images, type-B ultrasonic images, etc.) in a unified 3D environment, aligned to a 3D model of a human body. GSDM could also be used as a plugin for a 3D presentation tool such as Microsoft’s PowerPoint but in 3D. In this case, text and animation abilities should be further developed.

## 6. Acknowledgement

The work has been partially supported by the VISIONAIR project funded by the European Commission under grant agreement 262044, the French National project Co-DIVE and by the Italian National Project “Tecnologie e sistemi innovativi per la fabbrica del futuro e Made in Italy”

## Reference

- [1] G. Smith, “Idea-generation techniques: A formulary of active ingredients,” *Journal of Creative Behavior*, vol. 32, no. 2, pp. 107-133, 1998.
- [2] K. Sawyer, *Zig Zag: The Surprising Path to Greater Creativity*, Jossey-Bass, 2013.
- [3] A. Jain, T. Thormählen, T. Ritschel and H.-p. Seidel, “Exploring Shape Variations by 3D-Model Decomposition and Part-based Recombination,” *Computer Graphics Forum*, pp. 631-640, 2012.
- [4] X. Xie, K. Xu, N. J. Mitra, D. Cohen-Or, W. Gong, Q. Su and B. Chen, “Sketch-to-Design: Context-based Part Assembly,” *Computer Graphics Forum*, vol. 32, no. 8, pp. 233-245, 2013.
- [5] J. Lee and T. Funkhouser, “Sketch-Based Search and Composition of 3D Models,” *EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, 2008.
- [6] N. Jiang, P. Tan and L.-F. Cheong, “Symmetric Architecture Modeling with a Single Image,” *ACM Transactions on Graphics*, 2009.
- [7] El-Hakim and S. F., “Semi-automatic 3D reconstruction of occluded and unmarked surfaces from widely separated views” *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 34, no. 5, pp. 143-145, 2002.
- [8] L. D. Luca, P. Veron and M. Florenzano, “Reverse engineering of architectural buildings based on a hybrid modeling approach,” *Computers & Graphics*, vol. 30, no. 2, pp. 160-176, 2006.
- [9] M. Panchetti, J. Pernot and P. Véron, “Towards Recovery of Complex Shapes in Meshes Using Digital Images for Reverse Engineering Applications,” *Computer-Aided-Design*, vol. 42, no. 8, pp. 693-707, 2010.
- [10] D. Décriteau, J.-P. Pernot and M. Daniel, “Towards a declarative modelling approach built on top of a CAD modeller,” *Computer-Aided Design and Applications*, 2015.
- [11] N. Mitra, M. Wand, H. (. Zhang, D. Cohen-Or, V. Kim and Q.-X. Huang, “Structure-aware shape processing,” *SIGGRAPH Asia 2013 Courses*, New York, NY, USA, ACM, 2013, pp. 1:1--1:20.
- [12] G. Reeb., “ Sur les points singuliers d’ une forme de Pfaff compl`etement int`egrable ou d’ une fonction num`erique,” *Comptes-rendus de l’ Acad`emie des Sciences*, pp. 848-849, 1946.
- [13] S. Biasotti, D. Giorgi, M. Spagnuolo and B. Falcidieno, “Reeb graphs for shape analysis and applications,” *Theoretical Computer Science*, vol. 392, no. 1-3, p. 5 - 22, 2008.
- [14] C. Luciano da Fontoura and J. Roberto Marcondes Cesar, *Shape Analysis and Classification: Theory and Practice*, FL, USA: CRC Press, 2000.
- [15] I. Bloch, “Fuzzy relative position between objects in image processing: A morphological approach.,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 7, pp. 657-664, 1999.
- [16] C. Hudelot, J. Atif and I. Bloch, “Fuzzy spatial relation ontology for image interpretation,” *Fuzzy Sets Syst*, vol.

159, no. 15, pp. 1929-1951, 2008.

- [17] C. M. Takemura, "Modelagem de posições relativas de formas complexas para análise de configuração espacial," 2008.
- [18] AIM@SHAPE, "AIM@SHAPE," 2004. [online]. Available: <http://www.aimatshape.net/>.
- [19] "office room," [online]. Available: <http://www.decosee.com/2014/04/07/modern-office-room-minimalist-idea-23394.html>.
- [20] R. Vanderbei, *Linear Programming: Foundations and Extensions*, Springer-Verlag, 2001.
- [21] S. Mehrotra, "On the Implementation of a Primal-Dual Interior Point Method," *SIAM Journal on Optimization* 2, pp. 575-601, 1992.
- [22] J. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal* 7, pp. 308-313, 1965.
- [23] K. Price and R. Storn, "Differential Evolution," *Dr. Dobb's Journal* 264, pp. 18-24, 1997.
- [24] L. Ingber, "Simulated Annealing: Practice versus Theory," *Mathematical Computer Modelling* 18, vol. 11, pp. 29-57, 1993.
- [25] T. Tutenel, R. Bidarra, R. M. Smelik and K. J. de Kraker, "The role of semantics in games and simulations," *ACM Comput. Entertain.*, p. Article 57, 2008.
- [26] "Unity3D," [online]. Available: <http://www.unity3.com>.
- [27] Mathematica9. [online]. Available: <http://www.wolfram.com/mathematica/new-in-9/>.
- [28] F. B., G. F., L. J-C. and P. J-P., "Processing free form objects within a Product Development Process framework," *Advances in Computers and Information in Engineering Research*, J. G. Michopoulos, C. J. Paredis, D. W. Rosen and J. M. Vance, editor, ASME, 2014, pp. 317-344.
- [29] H. Istvan T., G. U. W. Martin, S. Olga, D. Loris, D. A. Raffaele and R. Graphitech, "Shape Semantics from Shape Context," 2004.

**Recent titles from the IMATI-REPORT Series:**

- 16-01:** *Optimal strategies for a time-dependent harvesting problem*, G.M. Coclite, M. Garavello, L.V. Spinolo.
- 16-02:** *A new design for the implementation of isogeometric analysis in Octave and Matlab: GeoPDEs 3.0*, R. Vázquez.
- 16-03:** *Defect detection in nanostructures*, D. Carrera, F. Manganini, G. Boracchi, E. Lanzarone.
- 16-04:** *A study of the state of the art of process planning for additive manufacturing*, M. Livesu, M. Attene, M. Spagnuolo, B. Falcidieno.
- 16-05:** *Calcolo di descrittori ibridi geometria-colore per l'analisi di similarità di forme 3D*, A. Raffo, S. Biasotti.
- 16-06:** *An appointment scheduling framework to balance the production of blood bags from donation*, S. Baş, G. Carello, E. Lanzarone, S. Yalçındağ.
- 16-07:** *From lesson learned to the refactoring of the DRIHM science gateway for hydro-meteorological research*, D. D'Agostino, E. Danovaro, A. Clematis, L. Roverelli, G. Zereik, A. Galizia.
- 16-08:** *Algorithms for the implementation of adaptive isogeometric methods using hierarchical splines*, E.M. Garau, R. Vázquez.
- 16-09:** *Feature curve identification in archaeological fragments using an extension of the Hough transform*, M.L. Torrente, S. Biasotti, B. Falcidieno.
- 16-10:** *Comparing methods for the approximation of rainfall fields in environmental applications*, G. Patané, A. Cerri, V. Skytt, S. Pittaluga, S. Biasotti, D. Sobrero, T. Dokken, M. Spagnuolo.
- 16-11:** *Persistence-based tracking of rainfall field maxima*, S. Biasotti, A. Cerri, S. Pittaluga, D. Sobrero, M. Spagnuolo.
- 16-12:** *Studio e sviluppo di componenti semantiche riusabili per la progettazione di serious game. Applicazioni all'area edutainment*, A. Repetto, C.E. Catalano.
- 16-13:** *Optimal-order isogeometric collocation at Galerkin superconvergent points*, M. Montardini, G. Sangalli, L. Tamellini.
- 16-14:** *Identification of patterns of repeated parts in solid objects*, L. Chiang, F. Giannini, M. Monti.
- 16-15:** *Identification of patterns of repeated parts in solid objects Part II*, L. Chiang, F. Giannini, M. Monti.
- 16-16:** *A machine learning based framework for mapping aesthetic properties to 3D free form shapes*, A. Petrov, J-P. Pernot, F. Giannini, B. Falcidieno, P. Véron
- 16-17:** *Conceptual design of shapes in Virtual Environments through the re-use of heterogeneous data*, Z. Li, F. Giannini, J-P. Pernot, P. Véron, B. Falcidieno