Automated Parametrization of Two-Dimensional Drawings

Alexander N. Inozemtsev, Dmitry I. Troitsky, Nikolay M. Pushkin, & Nataly S. Grigorieva
Tula State University
Tula, Russian Federation

Mark W. McK. Bannatyne
Purdue University
West Lafayette, Indiana, USA

Abstract

A theoretical foundation of parametric modeling suggests a mathematical model of automated parametrization of existing 2D drawings. Developments in the preparation of a self-learning parametric CAD system, and possible approaches to its implementation are discussed.:

Keywords: *Parametric, CAD systems, 2D, 3D, modeling, education, industry, design modification, dimension editing.*

1. INTRODUCTION

Studying the daily work experience of the design departments at several major industrial companies has allowed the authors to determine which procedures are most labor consuming and have to be automated in the first place. Our estimation shows that up to 80% of designer's work time is spent on modifying existing projects while only 10-15% of the time is reserved for developing original parts and assemblies. Moreover, in certain cases (e.g. manufacturing attachments design) the number of original projects is zero because the manufacturing equipment in use limits the variety of designs.

There is a certain difficulty in determining which project can be called an original one and which project is a mere modification. It appears that the only reasonable approach is based on the designer's work time: whether it is faster to modify an existing project or to make a new one from scratch. But it leads to the following problem: different CAD systems used by designers provide different degrees of automation in creating new projects and modifying existing ones [1].

2. EXISTING DESIGN MODIFICATION

There are two distinctive approaches to design modification. In the first approach there are no any

links between or limitations applied to the set E of geometric entities. Each entity E_i can be moved, scaled, rotated independently. This is the modification technique used in the most popular engineering CAD AutoCAD (according to CADalyst, 75% of all CAD systems worldwide are various versions of AutoCAD) as well as in the vast majority of similar "electronic drawing board" systems.

Another, much less used approach is known as parametric modeling. The designer creates the set R of relations between the geometric parameters of the entities. These relations greatly facilitate design modification since changing one entity may lead to appropriate changing of many other entities. As a result design time, by our estimation, can be reduced by a factor of 15..20. The currently available parametric CAD programs are basically T-Flex, SolidWorks and Pro/Engineer.

3. Problem Statement

The existing parametric CAD packages make the designer to perform the highly complicated task of developing an initial parametric model that is of generating both the E set and the R set. It causes the following contradiction: it is quite easy for the designer to create the E set but creating relations is really tricky because such a parametric model is a complicated abstract representation of a real object. We see it as one of the major reasons for relatively low usage of parametric CAD software in industry.

Analysis of this situation has produced the task of automated generation of the set R of relations from the set E of entities. For the designer a solution of this problem would mean the possibility of automatic conversion of an existing electronic drawing (e.g. made with AutoCAD) into a parametric model [2]. In industry they call it a transformation of a "dead" drawing into a "live" one. To solve the problem we have developed a mathematical model of parametric modeling, a sequence of automatic parametrization procedures as well as a working version of the software.

4. MATHEMATICAL MODEL

For doing parametric design it is essential to have a virtual model of a real object that preserves its features such as the shape and the position of its bounding elements. In 2D modeling these elements are lines and arcs, in 3D modeling these are surfaces. Note that in the proposed geometric model the dimensions of the bounding elements are not a part of the model itself but are treated as external parameters. Then a geometric model M of an object is:

$$M = \left\langle \vec{L}, \vec{A}, \vec{P}, \vec{R} \right\rangle$$

Where,

$\vec{L}$ is a vector of boundary elements descriptions;

$\vec{A}$ is a vector of auxiliary elements descriptions;

$\vec{P}$ is a vector that represents the object's contiguity graph (for example, as a matrix). It indicates the positions of the $\vec{L}$ elements on plane or in space. $\vec{R}$ is a vector of functional links between the descriptions of the bounding elements.

Since the result of design is not a virtual model of the object but a generation of an accurate drawing it is necessary to add auxiliary elements - dimensions, extra views, cross sections, text, hatching etc. - to the model. There are various ways of defining the elements of $\vec{L}$ and $\vec{A}$. Some of them are: Canonical equations of lines, circles, surfaces; table-interpolation geometry definition; set of R-functions [3]. In any case each element is fully described by a set of variables that can be both numeric (like the coordinates of a line's end) and string (drawing's annotation). The model (1) represents a set of real objects that have the same set of bounding elements and the same contiguity graph but different sets of dimensional parameters. For a manufacturing engineer it means a group of similar parts with similar manufacturing process. To identify a specific design in the model we introduce a *visualization function* $\Re$:

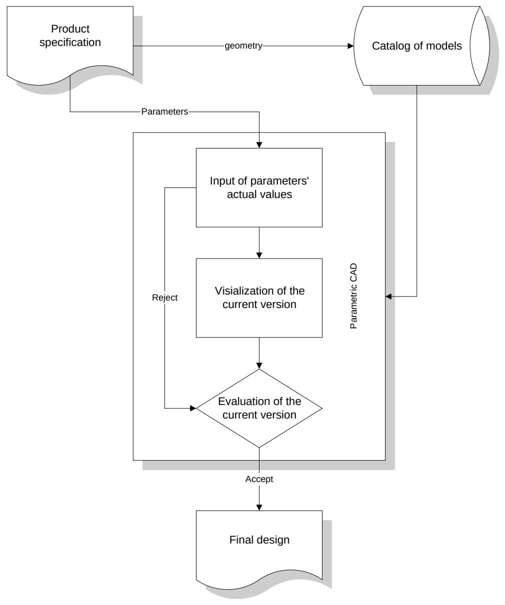
$$\Re\left(\vec{P}_0, \vec{S}, M\right)$$

Where,

$\vec{P}_0$ is a positioning vector that defines the position of the model in space;

$\vec{S}$ is a vector of dimensioning parameters of the model *M*.

The function renders a model by solving the system of equations stored in $\vec{L}$ and obtaining constant coordinates of all bounding elements. . Modification of the $\vec{S}$ components is parametric modeling. If we represent $\vec{S}$ as a point in a n-dimensional space M_f that embraces all variants of the model (*n* is the number of components in $\vec{S}$ then the search for the optimal design is a selection and testing of certain points in M_f until the design requirements are met.



Product specification
geometry
Catalog of models
Parameters
Input of parameters' actual values
Visialization of the current version
Reject
Parametric CAD
Evaluation of the current version
Accept
Final design

Figure 1: Parametric CAD's Flowchart

Design evaluation is performed by the designer in the form of a comparison of the function's (2) current result to the design specification. It should be noted that a design trajectory in the M_f space is generally a non-continuous one while the configuration of the M_f space limits the model and removes obviously impossible variants. As soon as the optimal variant is found the designer arranges a document (e.g. a drawing) by changing the $\vec{P}_0$ vector. This arrangement does not belong to design as such but is a necessary step in practice. Thus for parametric modeling of a certain class of objects it is sufficient to develop the model (1), to limit its dimensional space and to implement a CAD-designer interface being a loop with parameters' input, visualization, and evaluation (Figure 1).

5. PROCEDURES OF AUTOMATED PARAMETRIZATION

From now on we will deal with two-dimensional projections only since they are make up the majority of previously developed drawings. The set *E* of a drawing's geometrical elements can be divided into the following classes:

1. Profile elements that compose the contour of a projection;
2. Auxiliary elements that are completely defined by Class 1 objects. (e.g., centerlines);
3. Auxiliary elements that are partly defined by Class 1 objects (e.g. dimensions those origins are attached to corresponding profile elements while the dimension line position is arbitrary);
4. Independent auxiliary elements (cuts, cross-sections, extra views).

A parametric model includes the objects of Class 1 and Class 3. Class 2 objects can be rendered automatically by analyzing the profile (e.g. if there is an arc or circle we draw its centerlines). Class 4 objects require special treatment (see below).

At the first step we replace lines, circles and arcs of Class 1 with infinite construction lines. Construction lines provide links between profile's elements. Four construction lines define a line and an arc while a circle is defined by six lines (Figure 2). Note that is the proposed model a circle is defined by the center and by any point on its boundary. Besides all arcs are treated as circle's segments with constant radius. The rendering of construction lines is performed by profile's decomposition into lines and arcs and locating their endpoints and centerpoints that are trivial.

a. *At the second step* construction lines for Class 3 objects are added. Fig. 3 shows these lines that define the positions of dimension lines in various types of dimensions. As the two first steps are performed we have a set *CL* of construction lines.

b. *At the third step* an array *P* of intersection points of all *CL* lines is generated. We will call these points *base points*. The array contains actual values of coordinate in the world coordinate system.

c. *At the fourth step* the set E of Class 1 and Class 3 objects is generated. It can include lines, arcs and dimensions. A list of references to the P array's elements is created for each object. The list completely describes the geometry of the object. For instance, we have a line from (0,0) to (10,20). Suppose that in *P* the indexes of these points are 5 and 7 correspondingly. In this case the line in the *E* set would be represented as a list like (5 7 «LINE»).

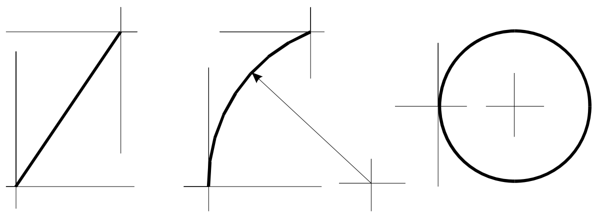


Figure 2: Lines, Arcs and Circles in Class 1

Along with the execution of this step the *correctness of the source drawing* can easily be checked. If a base point of an object does not belong to the *P* array then there is a violation of drawing's connectedness (e.g. the drawing is made with AutoCAD without proper usage of object snap) and the drawing has to be edited. After that the process of model's generation is over. For further use only two sets of data being *P* and *E* are saved. Their total size is minimal. Note that a presence of several views on the drawing does not make any drastic changes to the parametrization procedure. Moreover, construction lines automatically connect the views in the process of modification. Should a rotated view be present we have to consider *broken construction lines* instead.

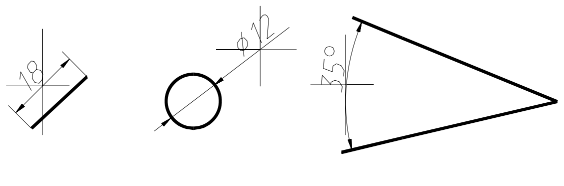


Figure 3: Construction Lines for Class 3 Objects

6. USING THE PARAMETRIC MODEL

A visualization function searches through the *E* set's elements, retrieves the corresponding actual coordinates of base points from P and renders the objects. Storing actual coordinates of base points eliminates one of the major problems with parametric modeling being the creation of a certain initial design "out of nothing" for further modification. In the proposed method the initial design is the same as the drawing used for making the model. Besides rendering the model's elements all the construction lines are displayed (like at the steps 2 and 3) and the base points are marked (like at the step 3). In a real implementation same code works in other cases. Finally we can proceed with parametrization as such. We offer three levels of parametrization:

a. *Shape level:* least accurate parametrization performed by moving the base points. It permits the definition of the general geometry of the object. A similar approach is available in SolidWorks 2000.

b. *Construction lines level*: more detailed parametrization performed by moving the construction lines. It corresponds to the technique implemented in T-Flex.

c. *Dimensions Level*: "classic" parametrization performed by providing actual value for a dimensional parameter.

d. Shape parametrization is done by changing the coordinates of a selected point within the *P* array and further re-visualization of the model. Certain difficulty arises when a point being moved is a start point or endpoint of an arc. To preserve the above-postulated constancy of the arc's radius the second point has also be moved to a position where the distances between the arc's center and both its start point and endpoint will be equal (Figure 4).

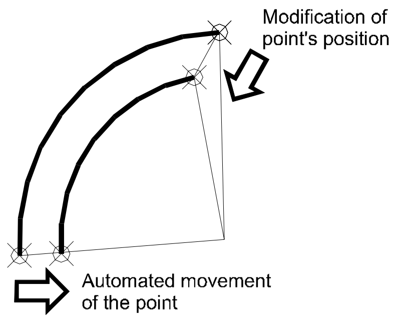

Modification of point's position
Automated movement of the point

Figure 4: Movement of an Arc's Start and End Points

Even more interesting case appears when the center point of an arc is moved. To avoid ambiguity we have postulated that out of two new radii the shortest one is maintained and the points are moved to keep it. When the center of a circle is moved the designer can naturally expect that the entire circle would be moved as well, not its diameter would be changed. That is why it is necessary to perform additional checking whether a point being moved is a center of a circle and if it is true to move the second point that defines the circle.

Construction lines parametrization includes finding the points in the *P* array that belong to the line being moved and re-calculation of their coordinates with the increment produced by a parallel movement of the line.

Dimensions parametrization is the most complex to implement. Firstly, there a re at least three types of dimensions: linear, radial/diameter and angular – that have to be treated differently. Secondly, changing a linear or an angular dimension can generally be done in three ways: by moving the first extension line, by moving the second extension line and by symmetrical moving of both lines (for radial/diameter dimensions we a priori postulate the immobility of the arc's or circle's center which removes ambiguity). That is why after entering a new value of a dimension the designer has to specify which of the three modification methods to use.

Dimension's modification should influence not only the base points which are the origins of its extension lines but also the construction lines passing through these points. This condition when applied to linear dimensions produces a certain difficulty with defining which one out of two construction lines that cross at the origin of the extension line being moved has to be moved. We have decided to move the construction line that makes a bigger angle with the dimension line. Such a rule works correctly for almost any linear dimension.

7. INTEGRITY OF DESIGN

A weak point of 2D parametric modeling has always been the possibility of designing impossible objects. For instance when a circle is moved the hole it represents may be located outside the part. Moving the base points may make the part's profile to cross itself. We must admit that in the existing parametric CAD system this issue is not considered at all. Our method provides a simple and easy-to-use solution to the problem.

After initial visualization (let us assume the initial model is correct) for each geometry's element E_i its number of intersections with other elements is memorized. Should the number of intersections in a new variant of the design be different from the initial one, the design has to be rejected as incorrect. For example a line on a part's outline has as a rule two intersection points with other objects. If we get three points we may conclude that the profile has crossed itself.

For better results the control dimensions method can also be used [4]. It means that a part of dimensions specified on the initial drawing are extra ones. They are marked in a certain way (e.g. by color). For each of the control dimensions the designer specifies its range. Should the actual value be out of range the system would produce a warning message. Obviously the designer does the generation of these ranges manually.

We have to keep in mind that one of the points being moved may happen to be a start point or an endpoint of another arc (e.g. the contour is made up of several conjugated arcs) and for preserving its integrity the procedure has to be performed several times.

8. DEPENDENCIES BETWEEN DIVERSE PARAMETERS AND SELF-LEARNING MODELS

One more important feature of parametric modeling is the presence of functional dependencies between model's parameters (the $\vec{R}$ set in (1)). These functional dependencies are implicitly implemented in T-Flex where the designer can manually enter quite complicated mathematical expressions for linking the parameters together. However our experience shows that generation of functional dependencies is unusual for designer's psychology and is considered to be tricky. Following the path of automation we have to provide a means for revealing functional dependencies of $f_{ij}(a,b)$ kind, where *i, j* are identifiers of the dimensional parameters that are linked; *a, b* are current values of these parameters. Note that the function has two arguments since it should work "both ways", that is, it should contain both direct and inverse dependencies.

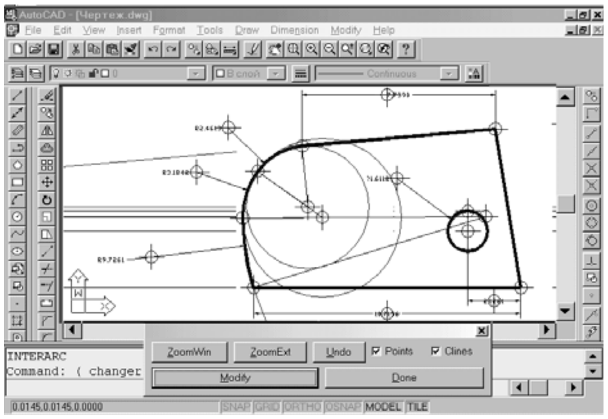
We could propose that in a vast majority of designs we would find simple linear dependencies like *kx+b*. To prove this hypothesis we have analyzed about 50 parts of automated rotor production lines (joint-stock company *Tula Cartridge Works*, Tula, Russia) and ball valves (joint-stock company *Tyazhpromarmatura*). Each part exists in a number of versions with different dimensions. No non-linear dependencies have been found. The deviations from linearity do not exceed 2%. That is why it is possible to suggest a concept of a self-learning parametric modeling system.

For self-learning the designer generates a number of designs while the dimensional parameters of each design are stored in a database. Then using a well-known mathematical technique of data fitting and approximation the system produces its own version of functional dependencies. The user decides whether these rules are correct or not.

The second stage of self-learning is limiting the produced dependencies. Generally a function generated by the system is non-continuous because some of its values may correspond to impossible designs. That is why for each function we have to define the area of its existence. To do it for each of the f_{ij} functions the system searches through the values of its arguments with a certain increment and generates "virtual" designs. If such design is incorrect the system marks a range of non-valid argument values. The range's boundaries may be later revised for higher accuracy. As a result we obtain a set $\vec{R}$ of dependencies that fully describe the real relations between the model's parameters.

9. IMPLEMENTATION

The authors have developed a working AutoCAD-based version of an automated parametrization system as Auto LISP application. It consists of a model generation module and a parametric design module. A self-learning module is being developed. Figure 5 shows a screen of the parametric model.


AutoCAD - [Чертеж.dwg]
File Edit View Insert Format Tools Draw Dimension Modify Help
Continuous
ZoomWin
ZoomExt
Undo
Points
Clines
Modify
Done
0.0145,0.0145,0.0000
SNAP GRID ORTHO OSNAP MODEL TILE

```
INTERARC
Command: ( changer
```

Figure 5: Implementation of the System

Depending on the entity selected by the user (a base point, a construction line, or a dimension) a corresponding kind of parametrization is invoked. The system generates standard .dwg-type electronic drawings that can be added to a catalog, printed out, included into an assembly etc.

10. CONCLUSION

The presentation of this paper offers a solution to a highly important industrial problem. Further research in this area is aimed at the implementation of an optimal self-learning algorithm. There are also many issues concerning complex drawings with rotated views, cross-sections, and hatching, as well as dealing with the complexities of working with entire assemblies. Our final goal is the development of a highly productive tool for converting existing legacy data into modern parametric models.

11. REFERENCES

[1] Grishin, S. A., & Dolgov, D. V.: Selecting a Proper CAD System for Manufacturing Attachments Design, *Proceedings of the Automation and Information Technology in Manufacturing International Scientific Conference (AIM'2000),* Tula, Russia, p. 103, 2000.

[2] Lyachek Yu. T., & Nakhimovsky Y. A.: Problems of Drawings Parameterization, *CAD (Computer - INFO),* No. 22, p. 164, 1999.

[3] Kutsenko L. N., & Markin L. V.: Shapes and Formulas, *Moscow Aviation Institute Press*, p. 176, 1994.

[4] Inozemtsev A. N., Troitsky D. I., & Bannatyne M. W. McK.: Parametric Modelling: Concept and Implementation, *Proceedings of the IEEE International Conference on Information Visualisation (IV'2000),* London, England. pp. 504-509, 2000.

About the authors

Dr. Alexander N. Inozemtsev is the Head of the Department of Automated Machine Tool Systems at Tula State University. Dr. Inozemstev began his career within the department as an Assistant Professor and rose quickly through the academic ranks to the position of Full Professor. He is widely respected for his work in manufacturing and the author of over 200 articles that deal with his field. Dr. Inozemstev has been the driving force for many new programs at Tula State University that have brought international recognition to himself and his department.

E-mail: **zem@uic.tula.ru**

Dr. Dmitri I. Troitsky is an Associate Professor within the Department of Automated Machine Tool Systems at Tula State University. His academic responsibilities include the development of computer programming and CAD courses, as well as working with diverse Russian industrial companies on the development of parametric modeling solutions for manufacturing applications. Dr. Troitsky has studied at university in the United States, and for several years was also the Coordinator of all international programs at Tula State University. He is the author of numerous published articles on manufacturing and has had his work presented at several international conferences outside of Russia.

E-mail: **troitsky@uic.tula.ru**

Dr. Mark W. McK. Bannatyne is an Associate Professor in the Department of Computer Graphics Technology at Purdue University. In addition to teaching solid and surface modeling, Dr. Bannatyne has also taught industrial machine tool applications. Dr. Bannatyne is an active member of Epsilon Pi Tau where he is a member of The Board of Editors for "The Journal of Technology Studies". Dr. Bannatyne's research agenda includes the problems faced by emerging nations in adapting technological solutions within education and industry and is the authors of numerous published articles on this subject. In 2000-2001, Dr. Bannatyne was a Fulbright Scholar in the Department of Automated Machine Tool Systems at Tula State University.

E-mail: **mwbannatyne@tech.purdue.edu**

Nataly S. Grigorieva entered Tula State University in 1995 and majored in "Industrial Automation". She received her B.Sc. with honors in 1999, and was a Masters student, 1999-2001 whne she graduated with honours. Miss Grigorieva's awards include the Medal of the National Student Research Contest (2000), and the Russian Universities Association Award (2000). She has also received numerous conference awards for "Best Report".

E-mail: **nata_ulcer@mail.ru**