FUNCTIONAL ASSEMBLY USING SYNAPTIC NETWORKS: THEORY AND A DEMONSTRATION CASE STUDY

Mavrikas, Georgios (1); Spitas, Vasilios (1); Spitas, Christos (2)
1: National Technical University of Athens, Greece; 2: Delft University of Technology, The Netherthands

Abstract

This paper proposes a methodology for the functional assembly of parts by creating synaptic networks between parts. Each synapsis contains both relative placement of part surfaces and features as well as the geometric tolerances that govern the functionality of the obtained part configuration. A case study is reported, where this method is seen in action and its merits are shown. The proposed method offers an integrated and concise approach to functional assembly modelling, by combining geometric placement of parts and tolerancing, which are normally treated separately by current CAD systems, with tolerances treated effectively as mere annotations, and largely neglected by current design theories.

Keywords: Design methods, Functional modelling, Tolerance representation and management, synaptic networks, functional assembly

Contact:
Georgios Mavrikas
National Technical University of Athens
Mechanical Engineering
Greece
mavrikasg@gmail.com

Please cite this paper as:
Surnames, Initials: *Title of paper*. In: Proceedings of the 20th International Conference on Engineering Design (ICED15), Vol. nn: Title of Volume, Milan, Italy, 27.-30.07.2015

1 INTRODUCTION

1.1 Problem definition

Since an awareness of the necessity of systematism in product and machine design initially appeared in the industry in the middle of the 20th century, engineers have been spending a great effort on creating tools in order to solve *design-for-X* problems, support or enhance creativity and optimise design parameters. Invariably, the product or machine being designed is a system, comprising several parts/ sub-systems, which by physically interacting with each other produce the designed system functionality. This design intent is embodied by means of assembling the various parts, thereby mating, joining and configuring them in space in a well-defined manner. It is therefore evident that core to the concept of assembly is obtaining functionalities, with the spatial placement of parts playing an ancillary –though important- role. Even more so, as the design process moves cyclically between abstraction and specification/ detail (Spitas 2011a, 2011b), it is even more critical to maintain a clear link between function and embodiment of the design intent. So how do we currently approach assembly systematically and how do we support this important process design-method-wise?

1.2 Design theories about assembly and standards for assembly tolerance annotation

Designers active in the industry and in the academia have created both problem-specific (to tackle various problems) and more general design methodologies and theories (Germani (2007), Koskela (2013), Russo (2013), Vermaas (2013), Blizzard (2013), Yanru (2014)). A quick survey of the literature shows that while in the various methods functionalities are tackled in some way and so is the geometric placement of parts, the functional aspect of spatial parts placement, typically communicated in the form of tolerancing, is missing.

Instead, the focus is more on problems like big data analysis and big data generation in the design process (e.g. in Majak et al. 2008) and problems referring to design decisions (Sheng et al. 2010), or different generic methodologies that, while conceptually robust and potentially usable, remain at a level of abstraction that has not yet integrated functional assembly issues like the ones discussed here (Clement et al. 2003, Pahl and Beitz et al. 2007, Shai et al. 2009, Albers et al. 2003). Despite the enrichments and the methodologies, the current design theories in their entirety (shown by Reich 1995), inadvertently pass by the problem of actual functionality in every product design problem. The lack of explicitly communicating functionality in the parts assembly process means that the designed assembly is actually not linked to its real function, but only to a record of the relative position of its parts.

Nonetheless, geometrical tolerances among working parts is a concept which exists in every assembly independently from their consideration in the design process. It is essential in the industry that the designer should demand some values of geometrical tolerances of the working surfaces in the assembly and address them on the technical drawings. For that reason, dimensional and geometrical tolerances have been extensively governed and classified by international standards, both for parts (ISO, 1989, 2011, 2012) and moveable assemblies (ISO, 2013). Despite the annotating rules and the classes, the standards are not filling the gap between annotating a drawing and functional tolerancing of the assembly. In other words, standards explain how to communicate/ annotate a tolerance, but not how to translate design intent into a tolerance, or backwards. In fact, there is ample anecdotal evidence in day-to-day industrial practice that tolerances are often copied from drawing to drawing and machine to machine, following an independently decided spatial placement of the parts, rather than systematically defined in each case and tailored to embody design intent. It is therefore meaningful, if not necessary, for the designer to use a method that helps him answer the fundamental question of "what is achieved by these tolerances" and guides him to wisely choose the tolerances that affect a specific function *while* deciding the spatial part placement. At the very least, such a method should help detect inconsistencies and enforce good tolerancing practice and proper embodiment of design intent.

1.3 Software programs for assembly and tolerancing

When discussing assembly support, one of course needs to consider the functionalities afforded by the CAD environments that designers use to perform assembly in the first place. The functionalities in

modern CAD programs (Solidworks, CATIA, Siemens NX, PTC Creo, Inventor 2014), in terms of tolerances, are limited up to the level of annotating within the program environment. All the possibilities that have been approved and standardized by the international standards are involved, but essentially the built-in tolerance support is merely an annotation-creation function with special symbols, lacking any functional connection to the tolerance parts/ assemblies, or any built-in logic. We list here just two of the possible (and frequent in the case on inexperienced engineers) misuses of this instrument while tolerancing assemblies:

1. The user is able to draw tolerances anywhere he desires (sketch, assembly, surfaces, lines), may lead to over-tolerancing the whole assembly, or even worse, tolerancing in a wrong way such as, according to the geometrical placements of the parts.
2. It is possible to attach a tolerance annotation to any surface or line without any constraint or warning from the program. Some examples, of acceptable tolerances, are cylindricity on flat surface, concentricity on a straight line and even circularity on a sketch dimension.

Definition of tolerances in CAD programs, are still very abstract and only for annotating use, without any connection between the meaning of a tolerance and the geometrical mate among parts. Söderberg 1995, 2013 and Bley 1996 have worked with tolerances in assemblies, in the directions of combined tolerance calculations, assemblies cost function minimization, designer's experience in CAP/CAT programs etc, taking for granted functionalities of the tolerances on the geometrical features. Therefore, there is a more fundamental gap to be addressed in CAD programs, where an integrated approach to spatial placement of parts and related tolerance definition could grant the designer the ability to functionally model the real interaction between parts.

1.4 Geometrical assembly vs. Functional assembly

So far in this introduction it has been made clear that both design methods and CAD environments lack a way to integrate the spatial placement of parts in an assembly and its tolerances that translate this placement into a functionality for a clear embodiment of the design intent. This paper proceeds to suggest such a method.

Firstly, this paper establishes a theoretical framework for object and relationship representation, based on unified Idea objects forming Synaptic Networks (Spitas 2013). Secondly, an algorithm is proposed, which is implemented in the form of an infinite loop running alongside the assembly design process, addressing such problems as have been identified previously in this introduction. While the paper does not focus on the development of a computer code to materialise this algorithm, any CAD software is ideal for such an implementation, as most of them are already equipped with assembly environments where the method can be implemented as a straightforward macro to extend the built-in functionality.

In prototype manufacturing and mould-making a standard practice is to finish-machine parts in the assembled state, in order to achieve closer assembly tolerances (in particularly co-axiality, parallelism/verticality, total runout) with less effort and cost than using stricter part tolerances, which would produce the same effect. Although this practice is uncommon for mass-produced parts/assemblies, it is still valid for small volume productions (CERN, 2014) and prototypes/moulds.

The whole idea is neither to replace the designer's key role in the assembly process by an automatic code, nor (in principle) to lighten his workload; but through this tool, the designer, will be able to create a functional assembly with meaningful tolerances, thus accomplishing two currently disconnected assembly steps in one step and avoiding omissions, errors and inconsistencies that would eventually cause a deterioration of the quality of the design and (if detected) additional workload.

Secondly, a demonstration is given of the workflow of the functional assembly algorithm on a real design case study. The goal of this demonstration is to make the reader familiar with the process and try to highlight the advantages of using this method, as encountered in practice.

This case study alone does not aim to validate any expectations in terms of performance gains in the design process. Creating a computerised tool and performing such validation in a controlled study will be the object of further research.

2 THEORY OF SYNAPTIC NETWORKS & ADAPTATION TO THE ASSEMBLY PROBLEM

In this work we adapt the generic Idea representations from Spitas (Spitas, 2013) to the specific problem of functional assembly relationships: Parts, part surfaces and features, the relative placements

of parts with regard to such surfaces and features, and the geometrical tolerances with regard to such relative placements (position, parallelism etc) are therefore all represented in the space of ideas, where each 'relationship' idea forms a synapsis connecting two or more 'part', 'surface' or 'feature' ideas.
Hereunder we show how a functional assembly is defined in terms of a synapsis, as opposed to a simple geometrical 'mate' (in CAD terms).
Let `PART(1)` and `PART(2)` be two simple orthogonal parallelepiped parts. Among other features, `PART(1)` has a face `SURF(1)`, a boss `SURF(2)` and another face `SURF(3)`, which are mutually orthogonal. Likewise, `PART(2)` has an orthogonal set of two faces `SURF(4)`, `SURF(5)` and a slot `SURF(6)`. These definitions can be seen in Figure 1. Thus far in idea object terms, the defined features may be represented in the format
`(feature-type (part) size-shape orientation-vector position-vector`[1]`)`

Table 1. Ideas definition format

name	Value
`SURF(1)`	`(FACE (PART(1)) PLANE ...)`
`SURF(2)`	`(BOSS (PART(1)) CYLINDER ...)`
`SURF(3)`	`(FACE (PART(1)) PLANE ...)`
`SURF(4)`	`(FACE (PART(2)) PLANE ...)`
`SURF(5)`	`(FACE (PART(2)) PLANE ...)`
`SURF(6)`	`(SLOT (PART(2)) PLANE ...)`

To place the parts in the context of an assembly (potentially involving more parts) we define synapses, each among two or more part surfaces, following the format `(synapsis-type (list-of-surfaces) definition`[2] `(list-of-tolerances))`, with each tolerance in turn following the format `(tolerance-type value material-condition`[3]`)`. The following table exemplifies one embodiment of an assembly:

Table 2. Synapses definition format

name	synapsis between	Value
`SYN(1)`	`SURF(1)` `SURF(4)`	`(MATE (SURF(4) SURF(1)) () ())`
`SYN(2)`	`SURF(2)` `SURF(6)`	`(MATE (SURF(6) SURF(2)) () ())`
`SYN(3)`	`SURF(3)` `SURF(5)`	`(FLUSH (SURF(5) SURF(3)) () ((PARALLELISM 0.01) (POSITION 0.02)))`

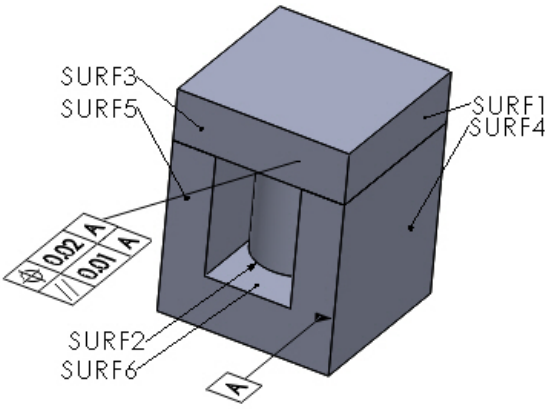


Figure 1. Definition of the paired surfaces and the necessary geometrical tolerances

[1] geometric definition in part coordinate system (size/ shape, orientation, position etc)
[2] relationships between orientation and position vectors, typically null if fully defined by the synapsis type
[3] if applicable

In the case of the first two synapses, which correspond to simple mating of plane surfaces (just like in CAD environment), there is no further functional requirement and hence no applicable explicit tolerances. In the case of the last synapsis, its non-contact nature means that its functionality must be secured by means of explicitly defined tolerances on the parallelism and relative position of the concerned surfaces. Clearly, a simple flush positioning of the same surfaces without the tolerance information would not suffice to communicate the design intent. By constraining the assembly in three directions (as the three synapses point out), the result is an explicitly determined assembly.
Thus it is shown that synaptic networks between parts can concisely contain and communicate all functional assembly information, including part positioning and tolerancing.

3 WORKFLOW AND ALGORITHM FOR FUNCTIONAL ASSEMBLY

The benefits of the proposed methodology are more obvious in the design cases where small/close assembly tolerances are required and less obvious in applications where higher tolerance fields are involved (e.g. PCB assemblies/electronics).
A CAD environment is an essential host for the described tool for some specific reasons: First of all, CAD/CAM/CAE programs are widely used in industry as a crucial part of the product manufacturing process, fact that brings the functional assembly tool in front of a big market with a well-known interest for continuous improvement of such software. Since in CAD programs, the engineers are modelling and simulating real structures, better implementations come on the surface. Thus these programs have been developing with a constant rate from the very beginning and nowadays provide user the capability of adding extensions to the CAD itself, through programming APIs.
Concerning the specific case of functional assembly implementation code, almost all CAD programs already contain an assembly environment, which can be used for the graphical implementation of the tool.
At the same with a graphical interface, the designer can also use the databases and the libraries of the program and so he can base on them, in order to build his own extension.
Last, but not least, for amplifying the consistency of the tool for functional assembly, among others, there are tools which will be part of the code because their features are part of the functional assembly logic.
Figure 2, depicts the workflow of the algorithm which actualizes the logic behind functional assembly. Initially, the designer has the choice to either assemble the parts by the “Mate” command, or by the superset of this command “Functional Assembly”. Choosing Functional Assembly (let be “FASSEM”), the code of this extension, starts running in parallel to the rest, while a synapsis has created in the synaptic network of this algorithm, waiting to semantically connect parts.
Algorithm’s first actual prompt, is for electing the essential parts, having the possibility to import new if not present. This prompt, has the meaning of adding into the working space, at least two parts which perform a certain function, and the designer desires to implement this function through his design. At any moment, the designer is able to add more parts in the working space. After this first step, the data that have been imported, will be added into a CAD default database and this is the reason for not including these data into Figure 2. In the space of the synaptic network, these essential parts, are single ideas, still not synapsed.
Being aware of the possible parts to functionally assemble, the designer is then impelled to define a certain positioning relationship between two parts, let be $PREL_{(I)}$, which is a synapsis between two parts-ideas. $PREL_{(I)}$ prompt, obligates the designer, to provide as input:

- a surface pair
- a vector pair and
- tolerance.

This prompt, has the meaning of forcing the designer to realize that the chosen surface pair, is the necessary and adequate pair for the specific function that he desires to achieve. This step, is a repeatable action and it is upon the designer to decide which surfaces from the space of all the essential parts, will be selected each time. The input after this prompt is this, which provides data to the extension database (depicted in Figure 2). The data from one PREL (let be “ASSEMBLYIDEA”), include:

- a pair of mates
- one or more geometrical tolerances and

- a physical contact flag (which is triggered from later steps).

Geometrical tolerances, is also a sub-group, which contains the data that define them explicitly:

- types of tolerances
- numerical values of tolerances
- material conditions and
- data references.

Consecutively, one positioning relationship and so the ASSEMBLYIDEA data, concern a single pair of surfaces.

Later in the algorithm, is the crucial step of deciding, whether the positioning relationship of the two surfaces, is at the same time, a physical contact relationship. This combination, defines the functionality. The decision taken, shapes the physical contact flag in the data of ASSEMBLYIDEA.

Without regard of the decision of the previous check, after many repetitions of $PREL_{(I)}$, a part has probably been synapsed with many others, alternatively meaning that probably, many surfaces of a part, have been positioned, oriented and toleranced at the will of the designer. The need for a second check, springs out of this complicated situation, which will shape the flag about the self-consistency of the part checked.

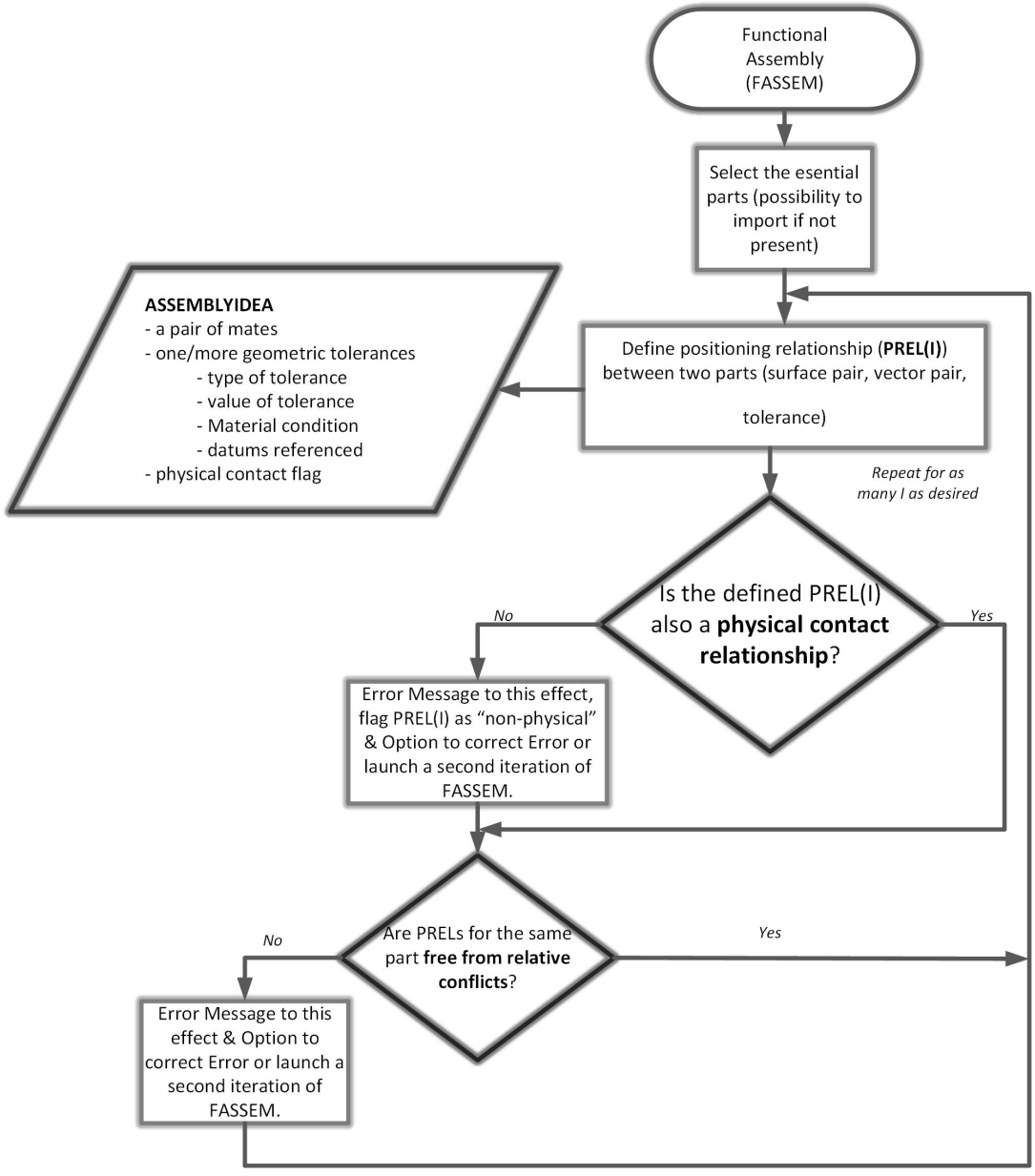


Figure 2. Flowchart of the algorithm for functional assembly

In a certain moment in this procedure, if all the positioning relationships, are also physical relationships and at the same time, all the parts are free from relative conflicts, then at that specific moment, what the designer has designed, is a functional assembly. At any time is it possible, that the designer changes some data, either to the direction of correcting the errors that appeared, or to the direction of continuing with another iteration passing by the errors. Far from drawing any bottlenecks in the algorithm, these degrees of freedom (designer's flexible actions), service the idea of iterative assembly, where the designer takes some decisions, receives the feedback and then improving his design towards that critical direction. For the described algorithm, the critical direction is functional assembly. In the end, the experience and the creativity of the designer on solving the conflicts, are the factors which will lead to a functionally improved assembly.

4 DEMONSTRATION CASE STUDY: "DIE FOR INJECTION MOULDING OF A THERMOPLASTIC BIOMEDICAL 4X6 PIPETTE BASE"

4.1 Context of the case study

The chosen demonstration case study, is the functional assembly of the mould core and cavity of a die for injection moulding of a thermoplastic biomedical 4x6 pipette base (Mavrikas et. al., 2014).
A plastic pipette, in biomedical terms, is used in order to transfer a fluid quantity into a biological sample. In this case, twenty-four flexible pipettes with internal conical geometry are connected to a plastic base with twenty-four external conical pipes, by sticking their engaged conical surfaces. A few of these assemblies, are later located on the top of a bigger grid of test tubes, while each hosting a part of a cartilage, in order to receive a specific protein from a robotic arm, through the base and the pipettes.
After specific time, the parts of the cartilage are tested for their new mechanical properties. The specifications of the design case, were the following:

- Polypropylene is the thermoplastic material (PP4052F).
- Hand injection moulding machine should be used.
- Generation of the taper external geometry at the endings of the pipes and the basic nominal dimensions of the base, with a certain deviation.
- Generation of the pipes conduits from the phase of injection moulding, is not obligatory

Basic issues of the injection moulding dies, which need to be faced:

- Precision alignment of core and cavity.
- Ventilation.
- Cooling.
- Metal contraction.
- Clamping force.
- Ejection of the mould

4.2 Functional Assembly implementation

The procedure for the demonstration of the functional assembly algorithm, has the following steps. The idea is to embody the effects that the algorithm draws to the designer, through his well-known design process:

1. Design the core of the mould and decide the parting surface for the two parts of the die (Figure 3). Then he design the two cavities and the two solid parts which will have to be later assembled (Figure 4).

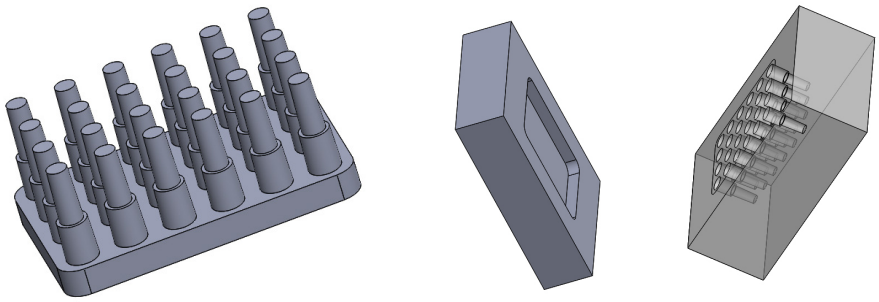
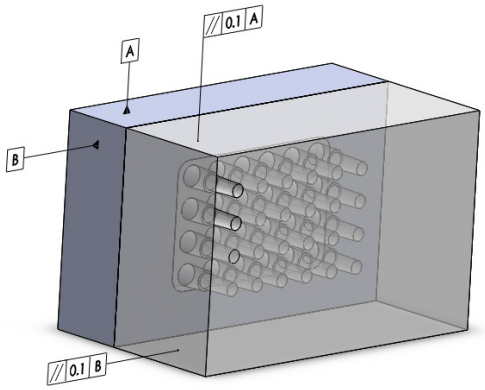


Figure 3. (left) Core of the mould.

Figure 4. (right) Parts of the die.

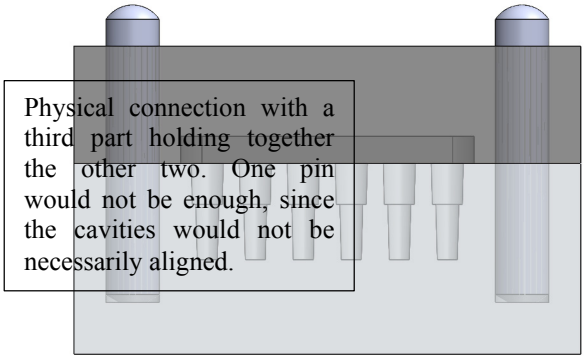
2. The first attempt for functionally assemble the parts, is the simple mate depicted in Figure 5 and it is failing the first check about physical relationship (Figure 5); which brings the designer to the condition where he should either correct the errors and come up with a design, passing this way the physical relationship check; or continue.



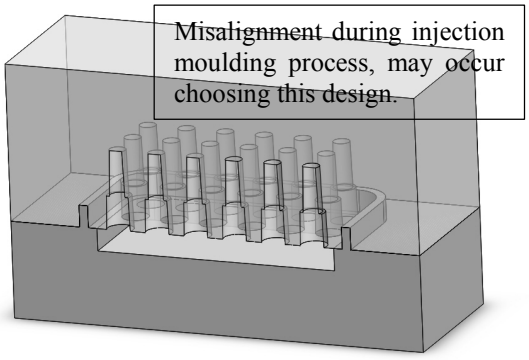
Mated surfaces which cause an error in the first functionality check. If the designer was choosing to continue, there is not a problem in the second check.

Figure 5. First non-functional assembly due to non-physical positioning relationships.

3. For the error correction about the non-physical contact, there is a space of design possibilities, but on this paper, only two are presented. The first alternative is to create some more female geometries which will be able to receive a male one (pin), so that the two main parts have now physical connection. On the other hand, the functional assembly is performed, by a pair of engaged geometries on the parting surfaces. The functional assemblies are shown below in Figures 6 and 7.



Physical connection with a third part holding together the other two. One pin would not be enough, since the cavities would not be necessarily aligned.



Misalignment during injection moulding process, may occur choosing this design.

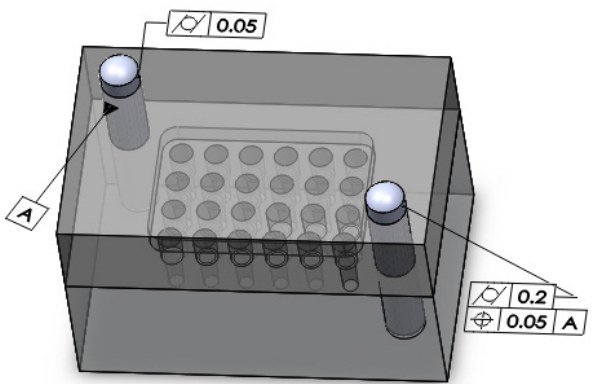
Figure 6. (left) Alternative design with dowel pins.
Figure 7. (right) Alternative design with a slot/boss formation on the parting surface.

4. The designer is facing the dilemma of choosing one alternative and this decision is out of reach of the functional assembly algorithm, if and only if, both alternatives are confirmed functional

assemblies. In this case, the decision was to use the alternative with the dowel pins, because the other geometry has a bigger alignment problem, as well as cumbersome air ventilation through the parting line.

5. On any attempt to design a functional assembly, the second check about self-consistency of a part itself, runs every time the designer either adds another PREL, or when he makes changes in the content of a current PREL (e.g. tolerance values). Figure 8 and 9 present examples of problems which causes the error warning for non-functionality, due to tolerances values conflict and vectors relations, accordingly. In the first case, calculations on relative tolerances can correct the error, while on the second, critical is the vectors fixing.

Finally, Figure 10 presents the real structure and the final product.



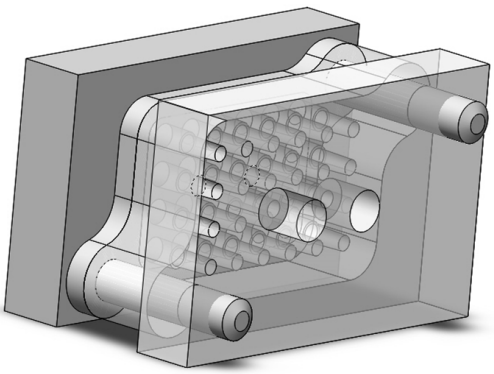


Figure 8. (left) Error in geometrical tolerances causing the second functionality check to fail.

Figure 9. (right) Assembly with problematic pins and parts vectors relations.

Figure 10. One part of the die and the final polypropylene pipette base

5 CONCLUSIONS

In this paper a methodology was described for the functional assembly of parts by creating synaptic networks between parts, which was demonstrated by means of a case study. The relative placement of part surfaces and features as well as the geometrical tolerances that govern the functionality of the obtained part configuration were described in an integrated manner by means of synapses (assembly ideas). It was shown that this integrated approach helps to guide design in an integrated manner, prompting the simultaneous consideration of part placement and tolerancing thereby leading to more rational and functional designs. This is an improvement in the context of functional assembly modelling, considering that

- tolerances are treated separately from part placement in current CAD systems, with part placement information being a functional part of the assembly description and tolerances treated as mere annotations, and
- tolerances are largely neglected by current design theories.

REFERENCES

Autodesk Inc., Inventor, http://www.autodesk.com/products/inventor/overview

Bley, H., Seel, U., Günther, K.G., (1996) Solving technical problems in assembly system's design, CIRP Annals - Manufacturing Technology, Vol. 45, No. 1, pp. 11-15

Blizzard, J. L., Klotz, L. E., (2012) A framework for sustainable whole systems design, Design Studies, Vol. 33, No. 5, pp. 456-479

CERN, (2014) ELENA magnets project, ELENA Solenoid General Assembly Drawing No. AD_MLNAF0001

Clement, K., Jordan, A., Vajna, S., (2003) The Autogenetic Design Theory – an Evolutionary View of the Design Process, ICED'03, Stockholm, Sweden, 19-21 August, Scotland: Design Society, 689-690

Dassault Systemes, Solidworks, http://www.solidworks.com/

Dassault Systemes, CATIA, http://www.3ds.com/products-services/catia

Germani, M., Mandorli, F., (2007) How to automate the geometrical tolerances inspection: A reverse engineering approach, CAT 2005, 10 April, Tempe, USA, CIRP: Arizona, 147-156

ISO (1989) 2768-1/2 General Tolerances - Part 1: Tolerances for linear and angular dimensions without individual tolerance indications

ISO (2011) 5459 Geometrical product specifications (GPS) - Geometrical tolerancing - Datums and datum systems

ISO (2012) 1101 Geometrical product specifications (GPS) - Geometrical tolerancing - Tolerances of form, orientation, location and run-out

ISO (2013) 17863 Geometrical product specifications (GPS) - Tolerancing of moveable assemblies

Kittel, K., Vajna, S., Hehenberger, P., Zeman, K., (2011) Product model of the Autogenetic Design Theory, ICED'11, Copenhagen, Sweden, 15-19 August, Scotland: Design Society, 309-318

Koskela, L. J., Ballard, G., (2013) The two pillars of design theory: Method of analysis and rhetoric, ICED'13, Seoul, 19-22 August, Scotland: Design Society, 21-30

Majak, J., Pohlak, M., Küttner, R., Eerme, M., Karjust, K., (2008) Artificial neural networks and genetic algorithms in engineering design, EngOpt 2008, Rio de Janeiro, Brazil, 01-05 June

Mavrikas, G., Besmertis, F., Nikolakis, M., Ntarouis, A. M., (2014) Design and construction of a die for injection moulding of a thermoplastic biomedical 8x12 pipette base, Technical Report, Laboratory of Systems Bioengineering, NTUA

Pahl, G., Beitz, W., Feldhusen, J., Grote, K.-H. (2007) Engineering Design: A Systematic Approach, 3rd edition. London: Springer

PTC Inc., PTC Creo, http://creo.ptc.com/

Reich, Y., (1995) Acritical review of General Design Theory. Research in Engineering Design, Vol. 7, No. 1, pp. 1-18Russo, D., Spreafico, C., Duci, C., (2013) On the co-existance of FBS andtriz for simplifying design process in an iterative way, ICED'13, Seoul, 19-22 August, Scotland: Design Society, 93-102

Shai, O., Reich, Y., Hatchuel, A., Subrahmanian, E., (2009) Creativity theories and scientific discovery: A study of C-K theory and infused design, ICED'09, Palo Alto, USA, 24-27 August, Scotland: Design Society, 263-274

Sheng, I., Lim, I., Kok-Soo, T., (2010) Eco-Efficient Product Design Using theory of Inventive Problem Solving (TRIZ) Principles, American Journal of Applied Sciences, Vol. 7, No. 6, pp. 852-858

Siemens Product Lifecycle Management Software, Siemens NX, http://www.plm.automation.siemens.com/en_us/products/nx/index.shtml

Söderberg, R., (1995) On Functional Tolerances in Machine Design, Göteborg, Chalmers University of Technology

Söderberg, R., Wärmefjord, K., Lindkvist, L., (2013) Tolerance Plugin Module in Integrated Design, Proceedings of the ASME Design Engineering Technical Conference, Vol. 3B

Spitas, C., (2010a) Analysis of systematic engineering design paradigms in industrial practice: a survey, Journal of Engineering Design, Vol. 22, No. 7, pp. 427-445

Spitas, C., (2010b) Analysis of systematic engineering design paradigms in industrial practice: scaled experiments, Journal of Engineering Design, Vol. 22, No. 7, pp. 447-465

Spitas, C. (2013) Beyond frames: A formal human-compatible representation of ideas in design using non-genetic ad-hoc and volatile class memberships and corresponding architecture for idea operators, ICED'13, Seoul, 19-22 August, Scotland: Design Society, 031-042

Vermaas, P. E., (2013) On the formal impossibility of analysing subfunctions as parts of functions in design methodology, Research in Engineering Design, Vol. 24, No. 1, pp. 19-32

Yanru, Z., Yuchu. Q., Meifa, H., Wenlong, L., Liang, C., (2014) Constructing a meta-model for assembly tolerance types with a description logic based approach, Computer-Aided Design, Vol. 48, No. 1, pp. 1-16