Computer -Aided Design and Applications

http://www.cadanda.com

Toward a Bottom-up Approach to Automate the Design of Cooling Systems for Injection Molding

Juan M. Jauregui-Becker[1], Hans Tragter[2] and Fred J. A. M van Houten[3]

[1]University of Twente, j.m.jaureguibecker@utwente.nl
[2]University of Twente, H.Tragter@ctw.utwente.nl
[3]University of Twente, F.J.A.M.vanHouten@ctw.utwente.nl

ABSTRACT

This paper presents a bottom-up approach to automate the generation of cooling systems for injection molding. The focus is set on layout design, as it is in this phase where the cooling concepts are generated. To do so, the layout design problem is decomposed in a top-down manner in both the functional domain and the physical domain. This results in a new problem formulation where elements at different abstraction levels are related by syntactic rules. Design solutions are generated by transforming lower abstraction levels into higher ones, which results in a bottom-up generation strategy. The results of implementing this method in SolidWorks® and C#® are presented as the method is being described.

Keywords: design automation, injection molding, cooling system.
DOI: 10.3722/cadaps.2009.447-459

1. INTRODUCTION

Injection molding is an important manufacturing technique for producing plastic parts. This technique consists of injecting a hot polymer into the impression of a mold. After injection, the plastic is cooled down to a solid state. This process is performed by a series of cooling circuits, which usually consist of cylindrical shaped channels drilled into the mold. The cooling stage is of great importance in the injection molding process, as it affects the productivity and the quality of the final part. Around 2/3 of the cycle time is usually spent in the cooling phase of the molding process [1]. An efficient cooling circuit design can considerably reduce the cooling time, and in this way increase the productivity of the injection process. Furthermore, severe warpage and thermal residual stress in the final product can be avoided by designing for uniform cooling.

The design process of cooling systems is divided into three successive processes [2]:

1. Preliminary design: accounts for determining possible locations of cooling channels in the vicinities of hot-spots. At this phase, channels do not constitute a circuit yet.
2. Layout design: consists of connecting previously positioned channels into circuits to assure a coolant can flow through it. Inlet and outlet channels are included to exchange the coolant with external cooling devices.
3. Detailed design: Optimizes cooling channel positions to minimize warpage and thermal residual stresses by applying small changes to the channel positions.

At present, most academic research in injection molding cooling design has focused on developing detailed analysis and optimization strategies. The former simulate the cooling phase of the injection

molding process and has resulted in Computer Aided Engineering (CAE) systems as SigmaSoft [3], MoldFlow [4] and Moldex3D [5]. The latter uses heat transfer analysis to minimize cooling time and temperature distributions of the plastic part. Several algorithms and heat transfer models have been develop for this purpose, some of which can be found in [6, 7, 8 and 9]. However, a human designer has to create the initial design from where CAE and optimization methods can be applied. In fact, optimizing cooling channels placement and diameter results in local optimum, given that the initial design bounds the possibility of finding global optima. Therefore, in order to achieve complete automation, the problem of generating the initial design must be solved.

Although automating design problems has been extensively researched over the past 30 years [10], not much literature addresses the problem of injection molding cooling design. The most relevant to this paper is the research developed by Li et al [11, 12, 13], which has resulted in a method for automating the design of cooling circuits for injection molding. In short, the method consists of decomposing the part geometry into several predefined shapes. Then, three techniques are collaboratively used for candidate solution generation, namely, case-based design, graph-based search and heuristic search. Case-based design maps the shape features to predefined solutions, obtaining a preliminary design that is captured in a graph model. A graph based transversal algorithm is employed to search for candidate cooling circuits. Heuristic search develops the candidate solutions into layout designs that contemplate tentative manufacturing plans. Although this approach has demonstrated to be capable of automating the generation of cooling solutions, it suffers from the drawback that the design has a strong dependency on the accuracy of the shape recognition algorithm as well as on the quality of the sub-solution predefined for each shape feature. Furthermore, complex algorithms are required to solve geometric constraints and keep the physical consistency of the cooling solutions (e.g. make sure cooling channels are connected to form a circuit).

In this paper a bottom-up approach to automate the generation of cooling systems for injection molding is presented. The focus is set on the layout design phase, as it is in this phase where the cooling concepts are generated. The approach consists in decomposing the problem into smaller problem chunks to reduce both the geometric and topologic complexity of the system. The decomposition is performed in a top-down manner for the functional domain and the physical domain. This results in four new problem formulations at different abstraction levels: (1) voxel mesh discretization, (2) design points, (2) cooling channels and (4) cooling circuit. Each abstraction level consists of several elements related by syntactic rules that describe the instantiation order of its elements. The decomposed system serves as map to determine the generation strategy, which consist of: (1) making the voxel mesh, (2) generating design points, (3) searching feasible cooling channels, and (4) assembling cooling channels into cooling circuits. Following this strategy results in a bottom-up generation approach, given that lower abstraction levels are recursively transformed into higher ones. The method has been implemented using C#® [14] as programming language and SolidWorks® [15] as CAD modeler. The algorithms used in each generation step are not further described here, as the purpose of this paper is to demonstrate the feasibility of using this approach to automate the design of cooling systems for injection molding. Furthermore, generation algorithms are still subject of current research. Preliminary results of generated cooling solutions allow concluding the method is successful in automating the design process of cooling systems for injection molding.

The rest of this paper is organized as follows:

- Section 2 starts by describing the framework used in this paper for structuring and modeling the design problem, followed by a summary of the variables and relations involved in injection molding cooling design. Also a brief discussion on the complexity is presented.
- Section 3 presents a top-down decomposition of the design problem into functional and primitive elements.
- Section 4 describes the four steps involved in the bottom-up approach to automatic cooling design generation.
- Section 5 presents an overview of the method, followed by the conclusions and recommendations.

2. COOLING CIRCUIT DESIGN

The problem of injection molding cooling system is structured and modeled here using the definitions in [16]:

- Element: is a class description of a design artifact component.

- Descriptions: characterize an element class by representing its attributes in the form of variables.
- Embodiment: is the subset of descriptions of an element upon which instances are created to generate design solutions.
- Scenario: is the subset of environment variables, attributed to elements in the natural world and considered in measuring a design artifact's ability to accomplish its function.
- Topology relations: define the configuration between embodiment and scenario elements.
- Physical coherence constraints: assure no physical impossibilities are committed by the artifact being designed.
- Performances: are those descriptions used to express and assess the artifacts behavior, and are calculated using analysis relations.
- Analysis relations: use known theories, for instance the laws of physics or economics, to model the interaction of the design artifact with its environment and predict its behavior.
- Objective function: weighs and adds the performances to result in the overall performance of the design.
- Confinement constraints: determine the range in which a description can be instantiated.

2.1 Problem Formulation

Fig. 1 shows the elements and relations involved in injection molding cooling design. The elements in the formulation are represented by nodes, while relations are represented by arcs. Labels are used to specify the models of the elements and relations. For explanatory reasons, not all descriptions and relations have been included. As the figure shows, Circuit is regarded as embodiment element, while Mold Part and Plastic Part are regarded as scenario elements. The goal of the design is to minimize the cooling time for the plastic part and to minimize the temperature differences in the plastic part. Cooling time and temperature distribution are therefore regarded as performances.

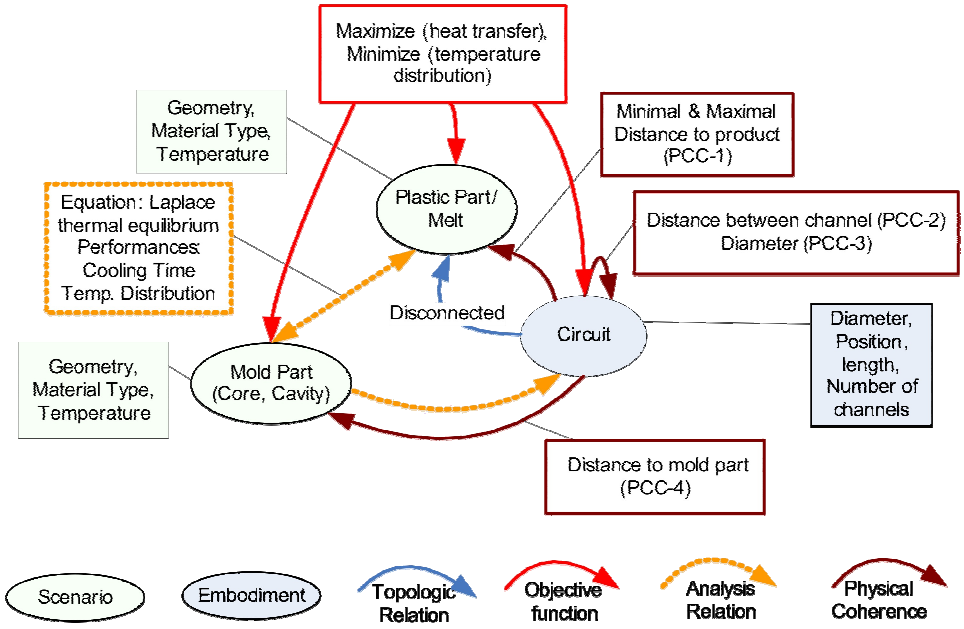


Fig. 1: Schematic representation of the formulation of cooling design problem.

Physical coherence, topologic and analysis relations are shown as arcs connecting the elements with labels specifying the relations. Analysis is governed by Laplace's equation, and calculates performance values: temperature distributions and cooling time. Detailed information on analysis methods is found in [1]. Topologic relations specify that elements such as Circuit are not allowed to share its space with

elements Plastic Part. Physical coherence constraints are used to define: (a) the minimum distance between a channel and the plastic part (PCC-1), (b) the minimum distance between channels (PCC-2), (c) the diameter values of the channels (PCC-3) and (d) the allowed distance between the channels and the Mold Parts (PCC-4). Designers often use empiric relations obtained by experience to approximate these distances and use them in a first design attempt. In Tab. 1, extracted from [2], these values are presented as a function of the thickness of the plastic part. According to [2], these values are considered a good starting point in the layout design process. Other physical coherence constraints specify non-drillable surfaces of the mold, surfaces where inlets and outlets can be placed and other mold parts (e.g. ejector pins and sliders) that constrain the space where cooling circuits can be placed. As the model is used here for explanatory reasons, these have been kept out of Figure 1.

Thickness (mm)	*Channel to Product distance (mm) (PCC-1, PCC-4)*	*Channel to Channel distance (mm) (PCC-2)*	*Channel diameter (mm) (PCC-3)*
0 to 1	11.3 to 15	10 to 13	4.5 to 6
1 to 2	15 to 21	13 to 19	6 to 8.5
2 to 4	21 to 27	19 to 23	8.5 to 11
4 to 6	27 to 35	23 to 30.5	11 to 14
6 to 8	35 to 50	30.5 to 40	14 to 18

Tab. 1: Experienced based physical coherence constrains for cooling system design.

2.2 Complexity Issues

Complexity in designing cooling systems arises from its large topology scale, complex geometries (both spatial and shapes) and high behavior interaction between its components, as indicated in Tab. 2. From the topologic point of view, the number of channels in one system dependents on the available space, the dimensions of the channel and the shape of the plastic part. Because of this, the number of channels in one cooling circuit varies largely from one mold to another. From the geometric perspective, the design is performed in 3 dimensions and usually involves complex geometries (e.g. double convex surfaces). At the same time, as molds contain many components (ejector pins, moving mechanisms, etc) the geometric space for placing channels is highly constrained. From the behavioral point of view, the heat transfer interaction between cooling system and mold is highly dependent on the number and disposition of the cooling channels. This makes it difficult to estimate the effect a particular channel has in the cooling process. However, as it is the differences in geometric shapes that require a unique cooling system for each part, behavioral complexity can be omitted from the design. In conclusion, a method to automatically design cooling systems for injection molding should be capable of managing both geometric and topologic complexity.

Aspect		*Cooling channel design*
Topologic (number of components)		10-1000
Geometric	Dimensions	Complex 3D
	Shape	Non uniform shapes
Behavior		Complex interaction
Functions (# number functional components)		3

Tab. 2: Complexity measures for injection molding cooling design.

3. TOP-DOWN COOLING PROBLEM DECOMPOSITION

Decomposing problems in different abstraction levels is often found in parameter estimation techniques, as for example in Structural Pattern Recognition (SPR) [15]. SPR is based on the assumption that objects can be modeled efficiently with hierarchically structured and attributed graphs. A graph

consists of a triple *V, A, F(V,A),* where *V* is a set of vertices or nodes, *A* is a set of arcs (also called edges, link or branches) and the function *F(V,A)* associates a pair of (unordered) vertices with each arc of the graph. Such models describe patterns by sets of primitive elements and primitive relations.

In this paper, top-down decomposition is performed with the aim of developing a model that –like in SPR– allows recognizing patterns that determine how to automatically generate cooling designs. Similarly, it is assumed that a hierarchical model can efficiently represent the design of injection molding cooling. The design is decomposed according to the guidelines presented in [18], where a method to manage complex domain integrated multidisciplinary routine design problems is described. The method consists in firstly performing a functional decomposition (Section 3.1) and secondly performing a primitive decomposition (Section 3.2). As shown in Section 3.3, the result is a new problem formulation consisting of functional elements related by topology relations and primitive elements related by physical coherence constraints. Relations are considered syntactic rules determining the instantiation order of each element in the formulation. This model is used in section 4 to present the strategy developed in this paper for automating the design of cooling systems.

3.1 Decomposition into Functional Elements

The functional decomposition is performed by identifying the functions involved in the artifact being designed. Then, elements undergoing more than one non-additive function (independent functions) are split into new element definitions: one element for each function.

Cooling circuits can be regarded as a collection of channels drilled into the core and the cavity of a mold. Cooling channels fulfill one the following functions:

- Function 1: cool the melt down. Channels are placed close to the part geometry and arranged such that heat is transferred in a homogenous manner from the melt to the coolant flowing through it.
- Function 2: transport the coolant. Coolant is transported between channels absorbing heat to constitute cooling circuits.
- Function 3: exchange coolant with the environment. Channels are placed at the surfaces of mold where the heat dissipation devices are connected to.

As these functions are not additive but complementary, the element channel can be decomposed into three independent elements: (a) Absorber channels to fulfill function 1; (b) Connector channels to fulfill function 2; (c) Exchanger channels to fulfill function 3. Fig. 3 shows the new problem formulation, which consists of three functional elements and three topology relations. According to this model, Connector channel can be connected to absorber channels. However, the relation does not work in the opposite direction. This is motivated by the fact that Connector channels function is constrained by the prior existence of an arrangement of Absorber channels. The same holds for the relation between Exchanger channels and the partial circuits formed by the arrangement of Absorber and Connector channels: the existence of the former is constrained by the prior existence of the later.

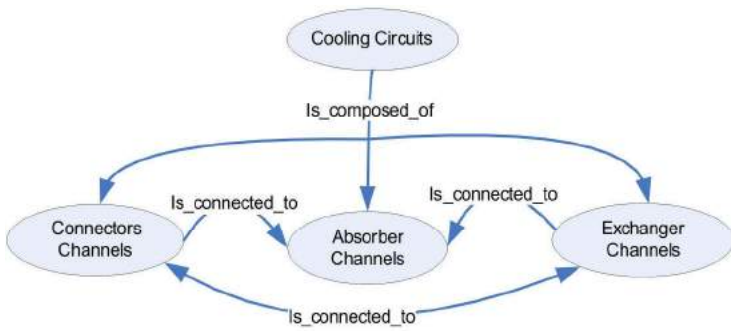


Fig. 2: Channels decomposed into functional elements.

3.2. Decomposition into Primitive Elements

Primitive decomposition is based on the design description classifications presented in [16]. Here, element attributes –either embodiment or scenario– are classified as five main types of models:

- *Parameters*: are used to model properties holding for the whole element. These can be of different nature, as for example numeric, symbolic, logic, predicate, and combinations among them.
- *Space*: Describe positional attributes of the elements in a topology. Positional descriptions depend on the chosen coordinate system –cartesian, cylindrical of spherical– and the dimensions of interest -1D, 2D, or 3D.
- *Field:* Use parameters and geometric vectors to describe properties that hold in specific regions of the elements. Fields are specified together with an incident zone, which is the spatial place where the field influences an element. An incident zone can be a volume, an area, a line or a point.
- *Shape:* Describes the form of the elements –or groups of elements– present in the design structure. Commonly used models are based on geometric relations and graphs.
- *Topology:* Describe how elements of a problem are related by topological relations as well as it models the number of times a specific element can be instantiated in the design solution. The latter property is regarded as the cardinality of an element.

Each category is regarded as a separate representational domain given that different reasoning schemes are required in each domain [16]. Here, primitive elements are constructed by indentifying the domains present in the problem and then encapsulating each information chunk into an independent element.

3.2.1. Channel Decomposition into Primitives

Fig. 3 shows the primitive decomposition. Channels are decomposed into primitive elements by analyzing the type of description used to model them. As the elements Absorber, Connector and Exchanger are modeled with the same descriptions, the decomposition is here generalized in an element channel which represents either one of the three before mentioned types. Channels have attributes in two domains, namely, shape and space.

The space descriptions are encapsulated into the primitive element Point. A Point contains a description about its position (*x,y,z*) in the cartesian coordinate system and a predicate description indicating its type (absorber, connector, Exchanger, etc). Each channel consists of two element Points: Initial Point (*Pi*) and Final Point (*Pf*). Points are related to the scenario elements by the physical coherence constraints represented by Tab. 1. Shape descriptions are encapsulated into the primitive element geometry, containing the description diameter (*D*). Having done this, the instantiation order of the primitive elements is determined by assessing the direction of the relations. This results in instantiating first the Points and then the Geometry. The Length is the distance between the initial and the final Point of one channel.

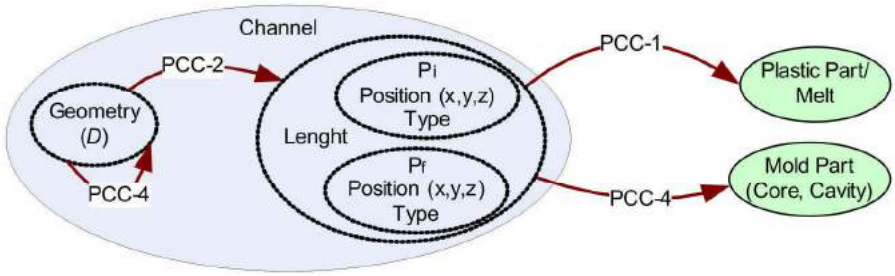


Fig. 3: Channel decomposed into primitive elements.

3.2.2. Mold Decomposition into Primitives

One difficulty in designing cooling systems, especially if it is to be done automatically, is reasoning about the spatial locations where the channels should be placed. Li et al [11] overcome this problem by

developing a library of standardized shape features and attributing to each feature a predefined channel layout. Having this, the method consists in recognizing shape features in the plastic part and mapping the predefined layout solutions. Latter, the channels are connected to form circuits using geometric and logic operations. This approach is also often used by designers.

This paper proposes a different approach, which consists of decomposing the mold parts –core, cavity and plastic part– into voxels elements. Voxels corresponds to the representational domain fields. Voxels are the 3D equivalent to pixels, and are commonly used by CAE programs to discretize solid parts for simulation purposes. In CAE systems, e.g. SigmaSoft® [3], voxels are used for calculating temperature distributions, deformations, cooling times, among others. In this case, voxels are described by its position in a grid of voxels and the mold type they represent.

3.3 The Resulting Problem

In Fig. 4, a directed graph shows the result of the problem decomposition. As shown, a hierarchical model consisting of three abstraction levels is obtained. As it is discussed in Section 4, by progressively transforming lower levels of abstraction into higher ones, the design of cooling systems can be performed in a systematic fashion, which allows its automation by a software tool.

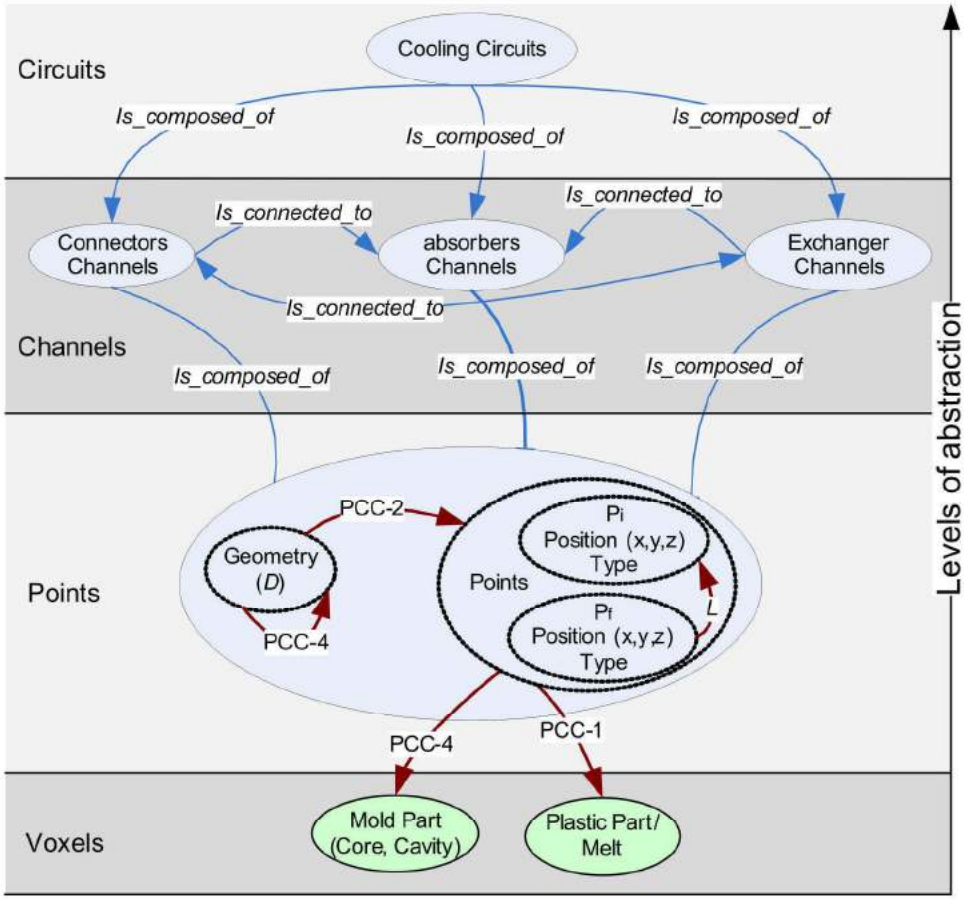


Fig. 4: Decomposed cooling design problem.

4. BOTTOM-UP COOLING DESIGN GENERATION

In this paper, cooling systems are automatically generated using a bottom-up approach. The term bottom-up is reserved for a strategy in which low level structures are transformed stepwise into higher

ones. By doing so, each design step focuses on solving one constraint or objective of the problem, which in this case correspond to the ones presented in Fig. 4. The method and algorithms for cooling generation have been implemented using C#® [14]. SolidWorks® [15] is used as interface for modeling the 3D mold parts. Furthermore, a User Interface (UI) was developed using SolidWorks® API. The UI allows users specifying characteristics of the mold, such as:

- Part type: define if a mold part is core, cavity or product, as shown in Fig. 5. Other part types (e.g. ejector pins) can be specified under the Optional Selection tab shown in Fig. 5.
- Surface type: defines if a surface can be used for placing inlets and outlets.

The underlying algorithms used for each generation step as well as the software implementation are not treated in this paper, as the purpose here is to present a general description of the method. Future publications will focus on this point. The following subsections describe the four steps of the method at the hand of a simple mold model of a telephone part, which is shown in Fig. 5.

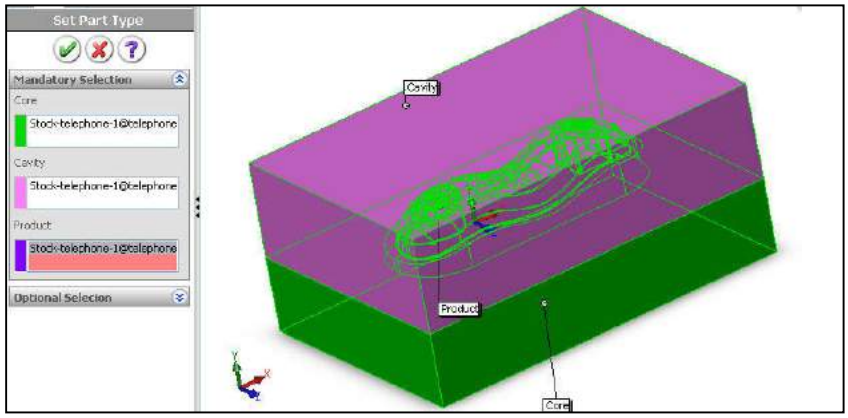


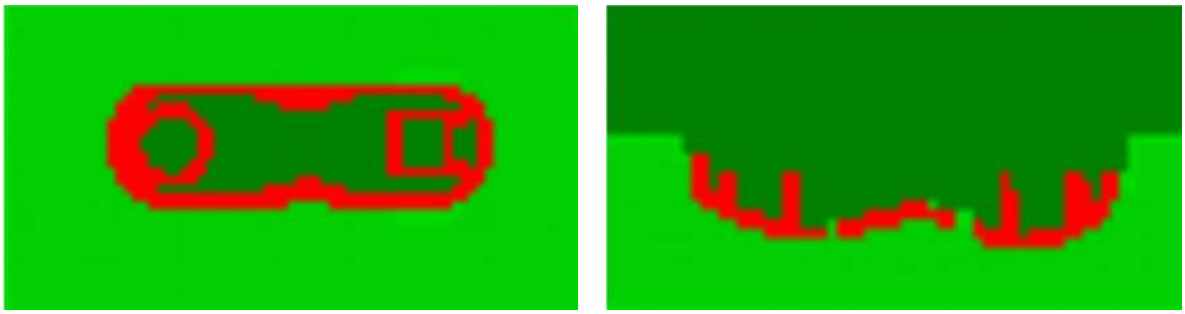
Fig. 5: Solid model of telephone.

4.1 Voxel Mesh Generation

Creating a mesh of voxels is the first step of this method. By using voxels, the geometric model can be transformed into a logic one. This idea is borrowed from Digital Image Processing (DIP), where image models are made as function of pixels. Here, pixels patterns are used to both generate new images or to recognize existing ones. Although this transformation results in a loss of geometric information, it also eases solving spatial constraints. Tab. 3 summarizes the attributes of a voxel object. Fig. 6 shows two sections of the voxel mesh of the telephone mold shown in Fig. 5. Here, the core is indicated in dark green, the cavity in light green and the part in red.

Attributes	*Description*	*Model*
Core	Identifies the core of the mold	Boolean
Cavity	Identifies the cavity of the mold	Boolean
Product	Identifies the product or plastic part of the mold	Boolean
Inlet/outlet	Identifies if a surface is used for inlets or outlets	Boolean
Non Drillable	Identifies if a surfaces cannot be drilled	Boolean
Size	Defines the size of the vertices of the voxel	Double
Position	Defines the position of the voxel	Double[i,j,k]

Tab. 3: List of attributes of a voxel.



(a) Horizontal section (b) Vertical section

Fig. 6: Telephone voxel mesh sections.

4.2 Points Generation

After voxel mesh generation, a 3D grid of points is generated. This grid is set inside the voxel mesh model of the mold. The distance between points in the grid is based on relation PCC-2 in Tab. 1. The objective of this step is to define the spatial locations in the mold where cooling channels can eventually be placed as well as those places where it cannot. To do so, a "color" is attributed to each point in the grid. This assigned attribute is used to characterize the function of the point. These are:

- Blue: defines points that can be used to create an Absorber channel.
- Green: defines points that can be used to create a Connector channels.
- Brown: defines points on the surface of a core and cavity where Exchanger channels can be placed.
- Grey points: define points nearby the melt where channels cannot be placed to overcome mold break.
- Black points: define points where channels cannot be placed to avoid mold break.

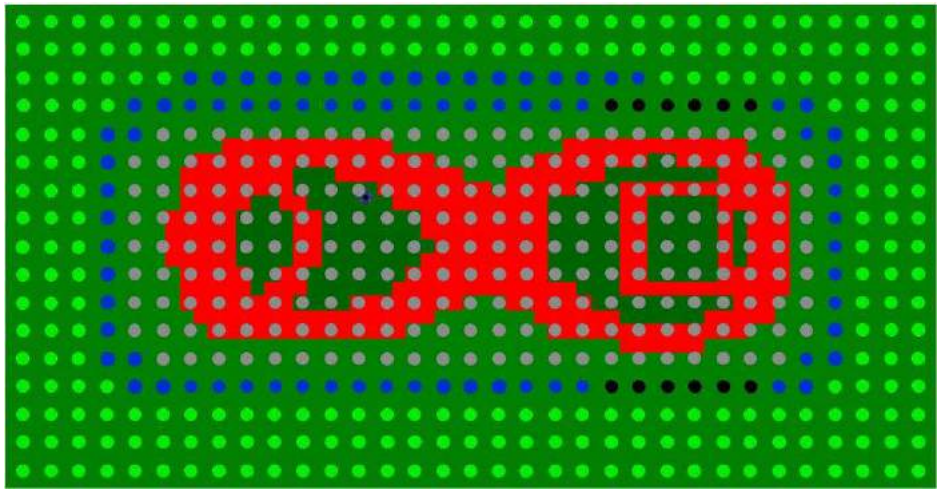
The color of a point is determined as a function of the voxels surrounding that point. In Tab. 4 the logic relations to determine the colors are presented. In relation to the decomposition shown in Section 3(a), this design step allows determining the primitive element Point for all three types of channels. Furthermore, this step determines the solution space of each design function in the problem, namely, to cool the plastic part (blue points), to transport the coolant (green points) and to exchange the coolant with exterior heat dissipation devices (brown points). Fig. 7 shows the result of applying the point grid to the case of the telephone mold.

Point color	*Surrounded by*
Green	(core voxels) or (cavity voxels)
Brown	[(core voxels) or (cavity voxels)] and (Exchanger voxels)
Black	[(core voxels) or (cavity voxels)] and [(non drillable voxels) or (product voxels))
Grey	[(core voxels) or (cavity voxels)] and (product voxels)
Blue	[(core voxels) or (cavity voxels)] and [Grey points]

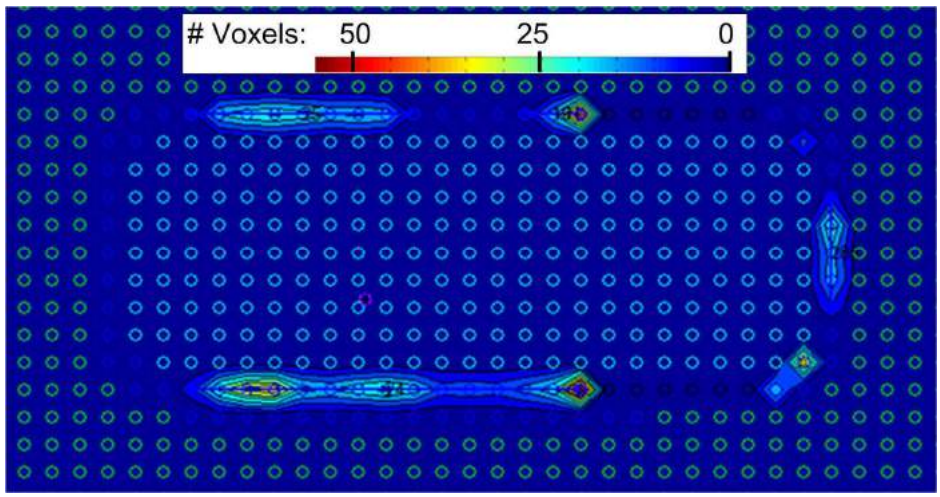
Tab. 4: Logical relations defining point's colors.

Additionally, the generated grid of points can be analyzed to measure the cooling performances (heat absorption and cooling homogeneity) at this abstraction level. A heat absorption index for each blue point is measured by counting the number of product voxels lying in the vicinity of a blue point. A cooling homogeneity index is calculated by dividing the heat absorption index by the sum of the distances from each product voxel to its closest point. Although such an analysis is not based on heat transfer principles, it provides a criterion for determining combinations of blue points to include in a cooling system. Furthermore, an optimization algorithm (e.g. simulated annealing) can use this

analysis to fine-tune the distance between points in the grid so that heat absorption and cooling homogeneity. Fig. 7(b) shows the results of calculating the heat absorption index of the section of the point grid in Fig. 3(a).



(a) Point mesh



(b) Analyzed point mesh

Fig. 7: Section of voxel mesh of telephone mold.

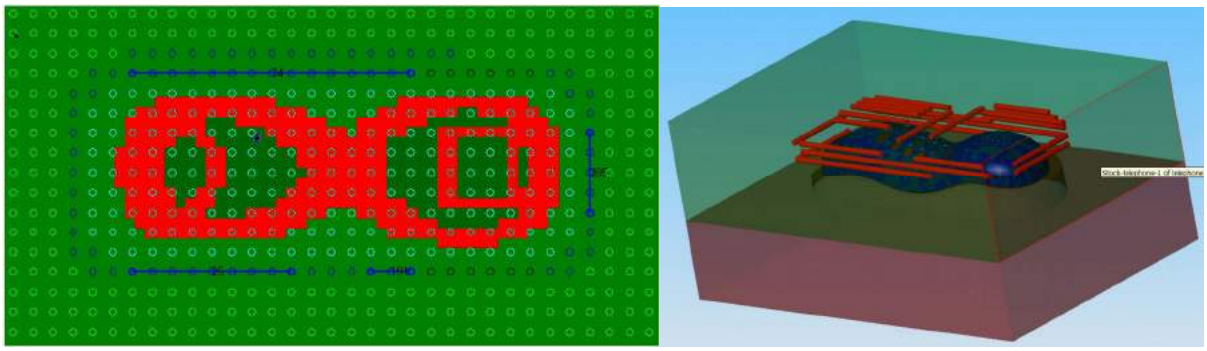
4.3 Absorber Channels Design

Section 3.1 concluded that the first channels to design are the Absorber channels. This is done in two steps. First, adjacent blue points are connected to form small segments, as shown in Fig. 8(a). Secondly, a search algorithm determines segment combination that can form channels by following a predefined criterion. The result of applying these steps is a list with all feasible Absorber channels.

The criterion used in the current implementation of the method is based on a drilling manufacturing technique, minimum channel length, minimum heat absorption index and minimum average homogeneity index. For example, the channels shown in Figure 8(a) and 8(b) have heat absorption indexes greater than zero (0) and lengths greater than one segment. As drilling was chosen for the channel's manufacture, only aligned segments are integrated into channels. However, conformal

cooling techniques can be integrated by defining new criteria on how to combine segments to form channels.

Furthermore, the manufacturability of each channel is assessed by determining if its prolongation is not constrained by black points. The diameter of the cooling channels is based on the values in Table 1 and is considered constant for all channels. One result of applying these steps can be seen in Figure 8(b), where absorber channels are shown for the case of the telephone mold.



(a) Section showing cooling channels (b) Non-overlapping cooling channels

Fig. 8: Absorber channels in the telephone mold.

4.4 Cooling Circuit Design

Cooling circuits are designed by assessing combination of Absorber channels capable of forming a circuit. A Genetic Algorithm (GA) [19] can be used to search optimal combinations of Absorber channels. Fig. 8.b presents a combination of Absorber channels that maximize heat absorption. The algorithm takes into account that two Absorber channels cannot share the same position in space. Once a set of Absorber channels is chosen, A* algorithm [20] is used to determine a path between them. This path determines the layout of the Connector channels, and therefore suffers the same manufacturability constraints as for Absorber channels. In the case a Connector channel crosses an existing Absorber or Connector channel, a new connection path is generated. By following these steps, several different circuits can be generated. Fig. 9 shows two cooling circuits for the case of the telephone mold. These results indicate the method can be used in automatically generating cooling systems for injection molding.

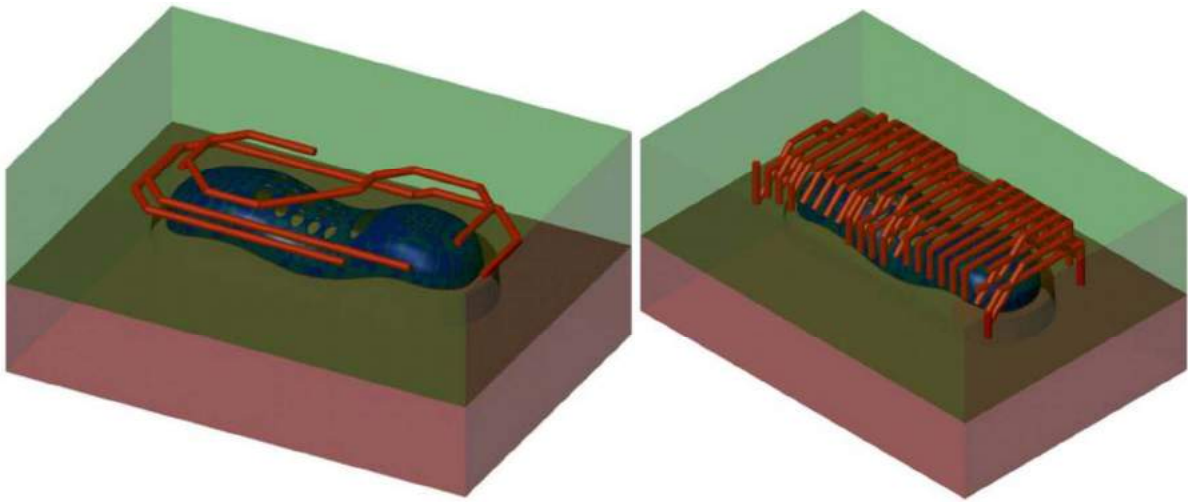


Fig. 9: Cooling circuits in telephone mold.

5. CONCLUSIONS

This paper presents a bottom-up approach to automate the generation of cooling systems for injection molding. The focus is set on the layout design phase, as it is here that cooling concepts are generated. Although the design of a cooling system depends on the thermal properties of the material of the mold parts as well as the geometric shape of the part, the method here presented is based on only on geometric considerations. This is done because it is the differences in geometric shapes that require a unique cooling system for each plastic part. Figure 10 presents a summary of the steps in the method.

In order to test the method, a software tool was implemented in C#. SolidWorks API was used to assist the user in defining the mold geometry as well as for presenting the cooling solutions generated by the tool. Preliminary results indicate the method is successful in generating cooling systems for injection molding. Further research will focus on:

- implementing a Genetic Algorithm that groups absorber channels to maximize heat absorption and cooling homogeneity,
- translating knowledge of expert mold designers to determine proper cooling channel combinations.

6. ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support of the Dutch Innovation Oriented Research Program 'Integrated Product Creation and Realization (IOP-IPCR)' of the Dutch Ministry of Economic Affairs. The authors especially want to thank PHILIPS ATC and in particular Fokke van der Veen for his effort, energy and contribution to this research project.

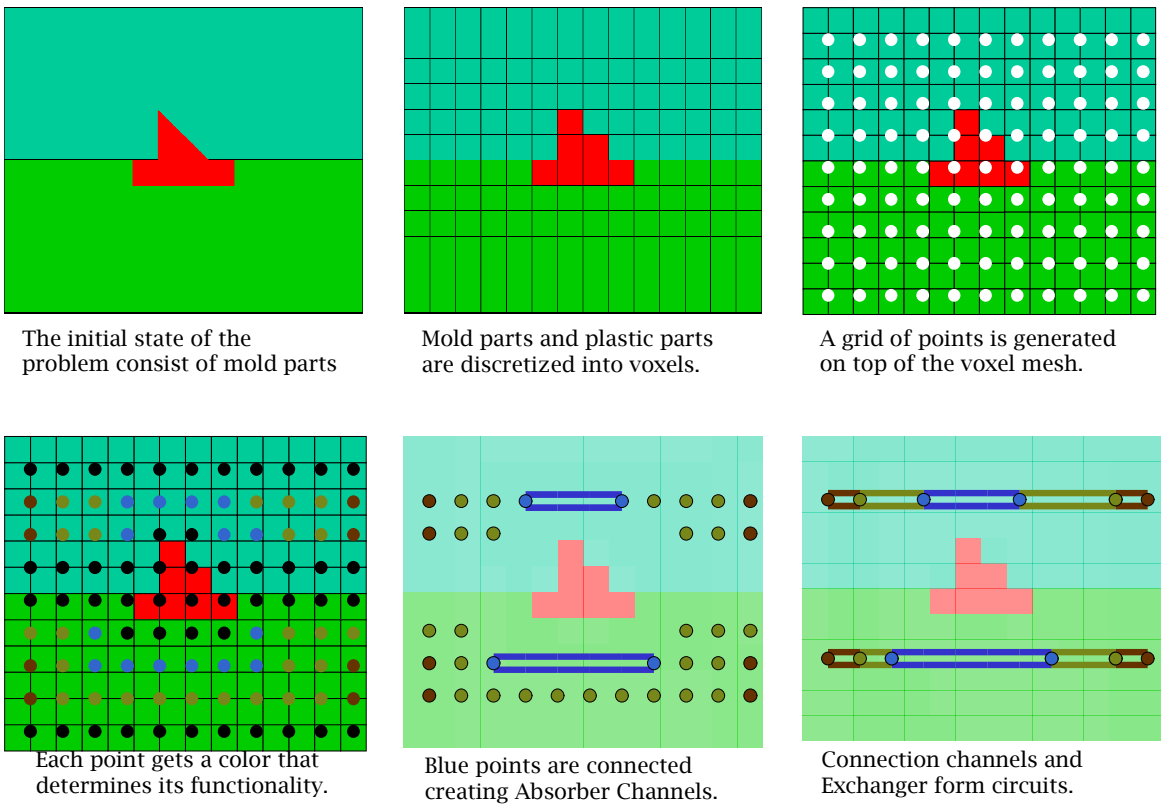


Fig. 10: Steps in the bottom-up approach to cooling circuit design.

7. REFERENCES

[1] Schumacher, R. N. S.: Design formulas for plastic engineers, Hanser Publishers, 2004.
[2] Menges, P.; Mohren, H.: How to make injection molds, Hanser Publishers, 1986.
[3] Kallien, H. L.; Steinbach, J.: Simulation of injection molding with 3D Volume Elements, Proceedings 18th CAD-FEM Users Meeting, volume 2, September 2000.
[4] MoldFlow, http://www.moldflow.com/stp/, Autodesk software.
[5] Moldex3D/Solid-RIM reference Manual, CoreTech System Co. Ltd., HsinChu, Taiwan, 2003.
[6] Park, S. J.; Won, D. T.: Optimal Cooling System Design for the Injection Molding Process, Polymer Engineering and Science, 39/9, 1998, 1450-1462.
[7] Mathey, E.; Penazzi, L., Schmidt, F. M.; Ronde-Oustau, F.: Automatic optimization of the cooling of injection mold based on the boundary element method, AIP Conf. Proc, 2004, 212-222.
[8] Pirc, N.; Schmidt, F.; Mongeau, M.; Bugarin, F.: BEM-based cooling optimization for 3D injection molding, International Journal of Mechanical Sciences, 48(4), 2006, 430-439.
[9] Rännar, L. E.: Efficient Cooling of FFF Injection Molding Tools with Conformal Cooling Channels - An Introductory Analysis, 1st International Conference on Advanced Research in Virtual and Rapid Prototyping, Leiria, Portugal, 2003, 433-437.
[10] Cagan, J.; Campbell M. I.; Finger S.; Tomiyama T.: Framework for Computational Design Synthesis: Model and Applications, Journal of Computing and Information Science in Engineering, 2005, ASME.
[11] Li, C. L.; Li, C. G.; Mok, A. C. K.: Automated layout design of plastic mold cooling system, Computer Aided Design, 37, 2001, 645-662.
[12] Li, C. L.; Li, C. G.: Manufacturable and functional layout design of cooling system for plastic injection mould - an automatic approach, In Proceedings of the international conference on manufacturing automation, 2004, 47-54.
[13] Li, C. L.; Li, C. G.: Plastic injection mould cooling system design by the configuration space method, Computer-Aided Design, 40(3), 2008, 334-349.
[14] Petzold, C.: Programming Microsoft Windows with C#, 2002, Microsoft Press.
[15] SolidWorks Corporation, SolidWorks 2007 API Documentation.
[16] Jauregui-Becker, J. M.; Tragter, H.; van Houten, F.J.A.M.: Structure and models of artifactual routine design problems for computational synthesis, CIRP Journal of Manufacturing Science and Technology, 1(3), 2009, Design Synthesis, 120-125.
[17] Pavlidis, T.: Structural Pattern Recognition, 1977, Springer-Verlag, New York.
[18] Jauregui-Becker, J. M.; Wits, W. W.; van Houten, F. J. A. M.: Reducing design complexity of multidisciplinary domain integrated products: a case study, In: Proceedings of the 41st CIRP Conference on Manufacturing Systems, CIRP MS 2008, Vol 1, Tokio, May 2008, 149-154.
[19] Whitley, D.: A Genetic Algorithm Tutorial Darrell, Statistics and Computing, 4, 1998, 65-85.
[20] Russell, S. J.; Norvig, P.: Artificial Intelligence: A Modern Approach. 2003, 97–104.