Spiral and Conformal Cooling in Plastic Injection Moulding

Yu Wang[1]* Kai-Ming Yu[1]† Charlie C.L. Wang[2]‡

[1]Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University

[2]Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong

Abstract

Designing cooling channels for the thermoplastic injection process is a very important step in mold design. A conformal cooling channel can significantly improve the efficiency and the quality of production in plastic injection molding. This paper introduces an approach to generate spiral channels for conformal cooling. The cooling channels designed by our algorithms has very simple connectivity and can achieve effective conformal cooling for the models with complex shapes. The axial curves of cooling channels are constructed on a free-form surface conformal to the mold surface. With the help of boundary-distance maps, algorithms are investigated to generate evenly distributed spiral curves on the surface. The cooling channels derived from these spiral curves are conformal to the plastic part and introduce nearly no reduction at the rate of coolant flow. Therefore, the channels are able to achieve uniform mold cooling. Moreover, by having simple connectivity, these spiral channels can be fabricated by copper duct bending instead of expensive selective laser sintering.

Keywords: conformal cooling, spiral channels, free-form shape, boundary-distance map, injection molding.

1 Introduction

As a common manufacturing process, plastic injection molding has been widely used to fabricate a variety of products. During a plastic injection molding cycle, the plastic part and the mold must be cooled to room temperature so that the molded part can be solidified and with its shape maintained. A substantial portion of the total molding cycle (e.g., as much as 80%) could be required for cooling. To improve the efficiency, cooling channels are usually integrated into the mold. In general, conventional cooling channels in simple shapes are fabricated by drilling straight-line holes. These usually lead to non-uniform mold cooling (ref. [1]). Without attaining the uniformity of surface temperature in a mold, the quality of plastic parts must be impaired by undesired defects, such as part warpage, sink mark, and differential shrinkage, etc. In addition, non-uniform cooling also increases the cooling time. In earlier studies [2, 3], effective cooling by using the conformal cooling system has been proved on parts with relative simple shapes. However, these approaches cannot be used for the products with free-form shapes, even after applying a feature-decomposition technique as proposed in [4].

1.1 Motivation

The work presented in this paper is motivated by automating the design process of conformal cooling channels for products with free-form shapes. Recently, we presented an approach in [5] to automatically generate circuit-like conformal cooling channels. The approach starts from offsetting the mold surface into a working surface, upon which a centroidal Voronoi diagram is used to help generate the cooling circuits. However, as the connectivity of a cooling circuit generated by [5] is complicated, the flow rate of coolant and also the temperature in the channels are highly non-uniform. Pumping expenses will thus have to be drastically increased to improve the efficiency of heat transfer and assure uniform coolant temperature. Furthermore, the fabrication of such cooling system with complex connectivity must be conducted by the additive manufacturing technique such as *selective laser sintering* (SLS), which is very expensive. Our new approach proposed in this paper aims at solving these problems by designing spiral cooling channels.

A number of factors must be considered while designing cooling systems for plastic injection molding, such as layout and connections of channels, composition of coolant, pressure drop of coolant and runner system, etc. In this work, we focus on the 3D shapes of conformal cooling channels. Specifically, we investigate algorithms to generate spiral conformal cooling channels so that heat transfer in the cooling system is optimized and fabrication costs are reduced. Similar to our prior work in [5], the axes of cooling channels are given on the working surface that is an offset of mold surface. Therefore, the shape of cooling channels is assured to be conformal to the mold surface. Uniform conformal cooling can be achieved as long as the temperature difference of the coolant between the inlet and the exit is small enough to be neglected. The efficiency of heat transfer is much higher in convention than conduction, and increases dramatically in turbulent flow. In particular, we focus on how to develop smooth spiral channels on the working surface conformal to the mold surface so that the turbulent flow is guaranteed.

*E-mail: carolyn_yuwang@hotmail.com
†E-mail: mfkmyu@inet.polyu.edu.hk
‡E-mail: cwang@mae.cuhk.edu.hk

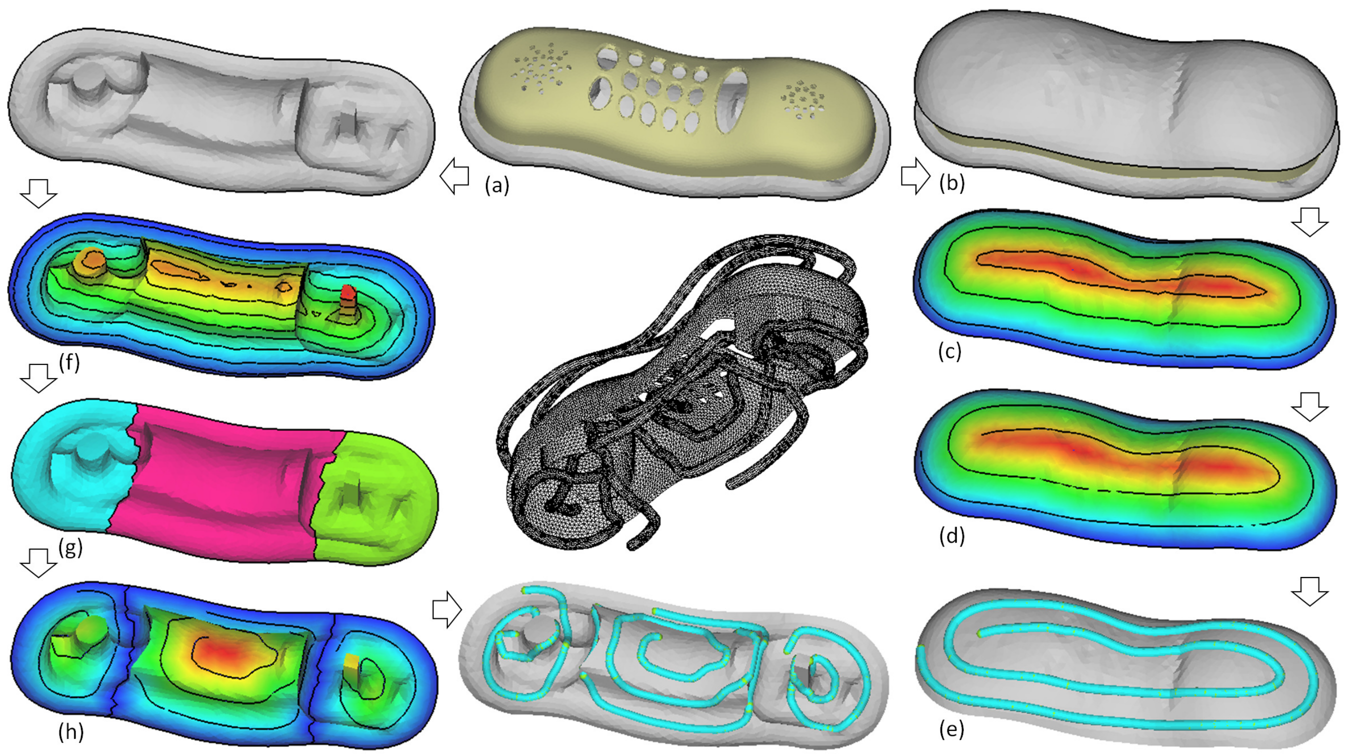


Figure 1: Overview of the pipeline for generating spiral and conformal cooling channels in the upper mold (right column) and the lower mold (left column).

1.2 Designing spiral and conformal channels

Our design methodology of spiral and conformal cooling channels can be illustrated by Fig.1. Given a cell-phone model to be fabricated by plastic injection moulding (see Fig.1(a)), offset surfaces are firstly constructed around it. The conformal cooling is accomplished by generating cooling channels on the offset surfaces. Part of the model's offset surface falling in the upper mold is used as the working surface for generating cooling channels for the upper mold (see Fig.1(b)). Taking this upper mold as an example, an enhanced Dijkstra algorithm is applied on the working surface to construct a piecewise linear approximation of the *boundary-distance map* (BDM) and its consequent iso-contours. Fig.1(c) shows the color map of its BDM and the iso-contours in black curves. The iso-contours are all in a simple topology (i.e., forming only one loop at a fixed iso-value). Our idea is to transform this set of iso-contours into a spiral curve with approximately even spacing (as shown in Fig.1(d)), in order to achieve uniform cooling. Finally, the spiral curves are served as axes to generate channels by sweeping a sphere along the curves (see Fig.1(e)).

However, the contours of BDM on the working surface of the lower mold are in more complex topology – see Fig.1(f), where some iso-contours have multiple loops. This brings in difficulty to generate a single spiral curve covering the whole working surface. To solve the problem, we develop an algorithm in this paper to first decompose the working surface into regions, such that each region is governed by a single spiral curve with nearly uniform space – see Fig.1(g) for an example. As a result, three spiral channels are generated on the working surface in the lower mold (see Fig.1(h)). In our approach, all the algorithms and computation are taken on the free-form surfaces represented by triangular meshes as shown in the middle of Fig.1.

1.3 Related Work

Designing and analyzing the conformal cooling channels for injection moulding have been studied for many years (e.g., [1–9]). The systems developed in [1, 2] involve a mathematical statement of the conformal cooling condition. Based on the criterion defined in [2], we developed a method to approximate the typical dimensions of cooling channels in our prior work [5]. This method will also be used to determine the dimensions of cooling channels in this paper. Many designers adopt the strategy introduced in [4] to design the final cooling system by synthesizing the sub-systems defined on each of the recognized features of plastic parts. However, as the feature decomposition in general is a hard problem, this strategy is difficult to be realized on molds with freeform surfaces. Alternatively, Park and Pham [3] proposes to decompose the regions according to the temperature distribution after the filling stage in molding simulation. Nevertheless, the computation of this approach may converge slowly on models with freeform shapes. Our region decomposition method presented in this paper is purely based on the geometric information – BDM, which can be computed efficiently. A recent effort to automate the design of cooling system is made in [9]. However, the channels in their work are designed in the zigzag shape, which can significantly reduce the flow rate of coolants.

In our work, all the channel axes are created on the offset surface surrounding the given model. This offset surface is assigned as the working surface. The grown offset surface of a solid model can be computed according to the mathematical definition given in [10]. Although the mathematical definition is compact, offsetting a freeform surface is not an easy job. We adopt the narrow-band signed distance-field (ref. [11, 12]) to generate the intersection-free offset surface for our cooling channels. Note that, the working surface must be intersection-free to prevent ill-topology on the axial curves of channels.

In the thread of research in CNC machining, spiral tool-path has been paid a lot of attention in the past (ref. [13–16]). Bieterman and Sandstrom [13] presented a method to use the solution of an elliptic partial differential equation (PDE) to morph a point (called center point) to the boundary of the shape. The spiral curves can only be generated on star-shaped polygons. In the work of Yao and Joneja [14], deformed Archimedean spirals are placed on the medial axis, and a few contour parallel offset curves are added near the boundary to connect all elements to a single tool path. To solve the problem of self-intersection and the generalization of shape to be processed, Held and Spielberger [15] investigated a method to generate spiral tool-path with the help of medial axis of a 2D polygon. None of these approaches consider the problem of generating spiral curves on free-form surfaces. Recently, a method is presented in [16] to generate iso-parametric tool-paths on surfaces represented by point clouds. However, only direction parallel tool-paths and contour parallel tool-paths are considered. In summary, an approach involving region decomposition for generating nearly-equidistant spiral curves on free-form surfaces remains an open problem.

1.4 Contributions

Our work has the following technical contributions.

- An efficient algorithm is developed to generate smooth spiral curves on free-form surfaces, where the spiral curves are governed by an approximated *boundary-distance map* (BDM) and have approximately uniform spacing.
- By analyzing BDM, a decomposition algorithm is investigated to segment free-form surfaces into regions that can be covered by contours of BDM with simple topology.

By incorporating the above two algorithms, a new design pipeline is investigated to generate spiral cooling channels for products with free-form shapes. Functionality of this approach will be demonstrated by experimental results and case studies.

Our paper is organized as follows. After introducing some preliminary terms in section 2, section 3 presents how to transform iso-contours of BDM into spiral cooling axes with even space. Section 4 describes BDM-based surface decomposition algorithm. Experimental results and case studies are shown in section 5. Finally, the paper ends with the conclusion section.

Table 1: Notation

Items	Description
l_A	The shortest distance from cooling channel wall to mold surface
l_B	The distance from cooling channel wall to the middle of two adjacent channels
l_m	Distance between the central lines of cooling channels to the mold surface
l_p	Half the plastic part thickness
W	Pitch distance between central lines of channels
D	Cooling channel diameter
ρ_m, ρ_p	Density of the mold and the plastic part
c_m, c_p	Specific heat of the mold and the plastic part
K_m	Thermal conductivity of the mold
h	Heat transfer coefficient
T_{melt}	Plastic melt temperature
T_e^A, T_e^B	Plastic ejection temperature at points A and B
t_{cycle}	Injection cycle time

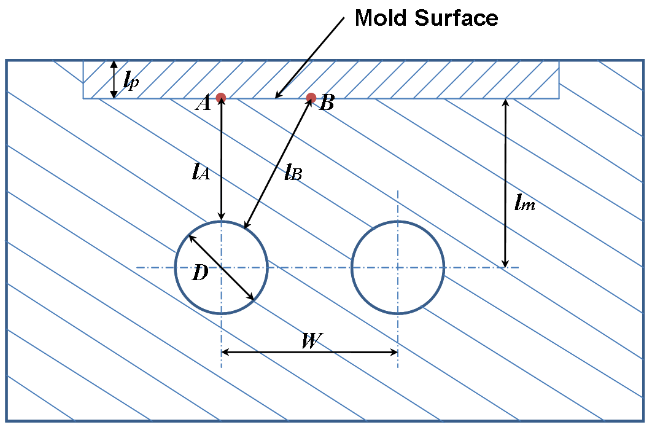


Figure 2: Typical dimensions of cooling channels in the heat transfer model, where D is the diameter of cooling channels and l_p is half thickness of a plastic part. Details of notation can be found in Table 1.

2 Preliminary

2.1 Physical model

This section briefly describes a method to use the thermal dynamic model to determine the geometric parameters of conformal cooling channels. More details about this physical model can be found in our prior work [5].

Considering a local cooling region in a cross-section of two adjacent cooling channels (as illustrated in Fig.2), a simplified formula for evaluating the temperature difference of mold surface at points A and B can be derived as

$$\overline{T_m^B} - \overline{T_m^A} = \frac{\rho_p c_p l_p}{t_{cycle} K_m}[(T_{melt} - T_e^B) l_B - (T_{melt} - T_e^A) l_A]. \quad (1)$$

The notation details are listed in Table 1. Uniform cooling can be obtained by controlling the mold temperature difference (i.e., $\overline{T_m^B} - \overline{T_m^A}$). According to the practical knowledge in cooling system design, the temperature difference should be within 10°C for producing parts with high accuracy – that is, $\overline{T_m^B} - \overline{T_m^A} \leq 10$°C. The two parameters, l_A and l_B, can be formulated by the typical dimensions for the cooling channel configuration as

$$l_A = l_m - \frac{D}{2}, \quad l_B = \frac{-D + \sqrt{4l_m^2 + W^2}}{2}, \tag{2}$$

where D is the diameter of cooling channel, W is the pitch distance between the axes of cooling channels, and l_m is the depth of cooling axes. To maintain a relative uniform temperature within the mold during an individual injection cycle, the value of l_m must be selected less than $\sqrt{t_{cycle}K_m/\rho_m c_m}$. However, the mechanical stiffness of a mold may be too weak to withstand a high pressure from putting cooling channels too close to the mold surface. As a result, the value of l_m is assigned as a value slightly smaller than $\sqrt{t_{cycle}K_m/\rho_m c_m}$ (e.g., with 10% reduction). Substituting Eq.(2) into Eq.(1) can lead to a value of W after choosing a diameter D of the channels according to the pump used in the cooling system. The rest of this paper will focus on how to generate the spiral curves with uniform spacing distance W on the working surface, which is a grown offset from the mold surface with distance l_m.

2.2 Boundary-distance map

Without loss of generality, we assume that the working surface used to generate spiral cooling channels is two-manifold and in the form of a triangular mesh. A triangular mesh M is usually represented as a complex $C = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V}$, $\mathcal{E}$ and $\mathcal{F}$ are sets of vertices, edges and triangular faces respectively. Information about the local connectivity, such as the left/right faces of an edge, the ordered edges inside a face, the edges linking to a vertex, is also stored together with the complex.

Definition 1 $\forall \mathbf{p} \in S$, $d_g^B(\mathbf{p})$ gives the geodesic distance from $\mathbf{p}$ to the boundary, ∂S, of the surface S as: $d_g^B(\mathbf{p}) = \inf_{\forall \mathbf{q} \in \partial S}\{d_g(\mathbf{p}, \mathbf{q})\}$, where $d_g(\mathbf{p}, \mathbf{q})$ denotes the geodesic distance from the point $\mathbf{p}$ to the point $\mathbf{q}$ on S.

Definition 2 A *boundary-distance field* is defined on every surface point, $\mathbf{p} \in S$, as $d_g^B(\mathbf{p})$.

The geodesic distance between two points on a differentiable surface can be evaluated by the first fundamental form [17]. However, it is impractical to compute a boundary-distance field in this way. Our algorithms presented in this paper are based on a discrete version of the boundary-distance field.

Definition 3 A *boundary-distance map* (BDM) is a set of scalars defined on the vertices of a triangular mesh M such that the scalar, $\bar{d}_g^B(\mathbf{v})$, at a vertex $\mathbf{v}$ specifies the approximate geodesic distance to the boundary of M.

The prior research in the computer graphics community has investigated efficient techniques for computing the exact/approximate geodesic distances on piecewise linear surfaces (ref. [18–21]). Here, we adopt a simple approximation akin to [22] to evaluate BDM on the triangular mesh M, which works well on triangular meshes with relative regular triangles.

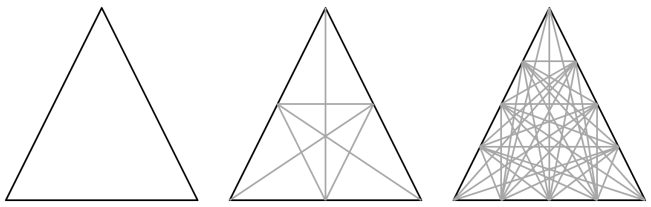


Figure 3: An illustration about adding virtual pathes (in gray) into triangles: (left) a given triangle, (middle) a 1-refined triangle with six virtual paths inserted, and (right) an example of 3-refined triangle.

First, virtual paths are constructed on each triangles by refining each existing edge – a graph consists of vertices and edges of M and the virtual paths is called k-refined graph of M if each edge on M has k virtual nodes inserted. Examples of different k-refined graph on a triangle is shown in Fig.3. Then, using graph nodes on the boundary edges of M as sources, Dijkstra's algorithm can be applied on the k-refined graph of M to compute a more accurate approximate of BDM. Obviously, when $k \to \infty$, the approximation converges to the exact geodesics but with the cost of computing time. We use 3-refined graphs in all the examples presented in this paper.

Definition 4 For a triangle T with three vertices $\mathbf{v}_i$, $\mathbf{v}_j$ and $\mathbf{v}_k$, the value of BDM at $\mathbf{p} \in T$ is defined as $\bar{d}_g^B(\mathbf{p}) = \alpha \bar{d}_g^B(\mathbf{v}_i) + \beta \bar{d}_g^B(\mathbf{v}_j) + (1 - \alpha - \beta)\bar{d}_g^B(\mathbf{v}_k)$ with (α, β) being the barycentric coordinate of $\mathbf{p}$ in T.

By this, BDM is defined throughout the working surface M, which is employed to govern the decomposition and the spiraling in the following sections. Note that, all the computations in the rest of this paper are taken on this piecewise-linear representation of BDM, $\bar{d}_g^B(\mathbf{p})$. The error analysis between exact and approximate geodesic distance has been taken in prior research (ref. [19, 20]), which is beyond the scope of this paper. In short, when the number of triangles on M goes to infinity, $\bar{d}_g^B(\mathbf{p}) \to d_g^B(\mathbf{p})$.

3 BDM-Based Spiral Channel Generation

In this section, we present how to generate spiral curves from the iso-contours of BDM with equal distance – i.e., W for the generation of conformal cooling channels. The topology of surface region we are working on is assumed to be ω-simple.

Definition 5 For the BDM of a surface M, if all iso-contours generated by the stepwise threshold $i\omega$ ($\forall i \in \mathbb{Z}$) have only one loop, the topology of surface M is named as

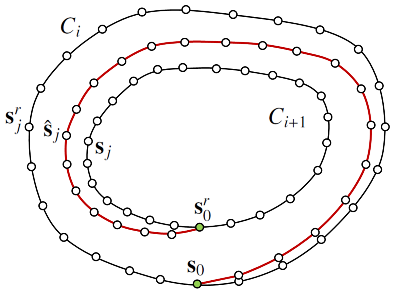


Figure 4: BDM guided spiraling – an illustration about how to spiral two iso-contours: C_i and C_{i+1}. The starting point for spiraling, $\mathbf{s}_0$, and its reference point, $\mathbf{s}_0^r$, are displayed in green. The red curve is the result of spiraling.

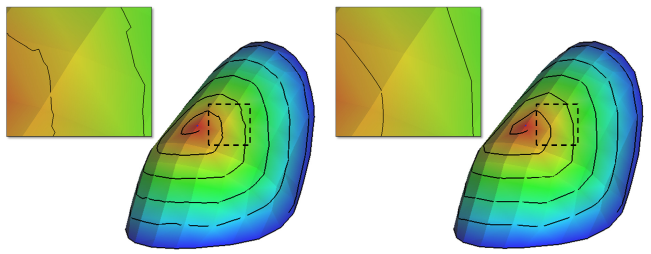


Figure 5: BDM guided blending for spiraling: (left) zigzag distortion can be caused by the error of geodesic approximation, and (right) smooth spiraling is generated by our dragging algorithm.

ω-simple in terms of BDM.

3.1 Spiraling between contours

Without loss of generality, we focus on spiraling between two iso-curves $C_i = \{\mathbf{p} \mid \bar{d}_g^B(\mathbf{p}) \equiv i\omega\}$ and $C_{i+1} = \{\mathbf{p} \mid \bar{d}_g^B(\mathbf{p}) \equiv (i+1)\omega\}$ below. Spiraling between other two neighboring iso-curves can be realized in the same way.

- First, given a point $\mathbf{s}_0$ on C_i to serve as the starting point for spiraling, its closest point on C_{i+1}, $\mathbf{s}_0^r$, is found. This pair of points will be used as the starting points for spiraling from C_{i+1} to C_{i+2}.
- Second, C_i and C_{i+1} are sampled into n points uniformly as $\mathbf{s}_j \in C_i$ and $\mathbf{s}_j^r \in C_{i+1}$ $(j = 0, \cdots, n-1)$ – see the illustration shown in Fig.4. The value of n is determined by $\max\{72, \|C_i\|/l_{avg}\}$, where $\|\cdot\|$ denotes the length of a curve and l_{avg} is the average length of edges on M. Note that the number of samples used for spiraling between different iso-curves could be different.
- Third, the set of points, $\hat{\mathbf{s}}_j = \xi(\mathbf{s}_j, \mathbf{s}_j^r, j/n)$, are generated by blending $\mathbf{s}_j$ and $\mathbf{s}_j^r$, where the function $\xi(\cdots)$ defines the way of blending with the parameter j/n. Linking the blended points, $\hat{\mathbf{s}}_j$, consecutively forms the spiral curve between C_i and C_{i+1}.

Repeating these steps can generate all the spiral curves between iso-curves of a BDM on the surface with ω-simple topology.

Now we discuss how to define the blending function, $\xi(\cdots)$. The simplest one is linear interpolation between $\mathbf{s}_j$ and $\mathbf{s}_j^r$ as: $\xi(\mathbf{s}_j, \mathbf{s}_j^r, t) = (1-t)\mathbf{s}_j + t\mathbf{s}_j^r$. However, a point generated in this way could run away from the input surface when the region between $\mathbf{s}_j$ and $\mathbf{s}_j^r$ is highly curved. As a result, the spiral curves generated in this way are not located on the surface. A more sophisticated method needs to be developed for blending two points on the surface, and the blending is governed by BDM.

Table 2: Location Detection by Barycentric Coordinate

(α,β)	Location in the triangle $\mathbf{v}_1^f\mathbf{v}_2^f\mathbf{v}_3^f$
$\alpha+\beta = 1$ and $\beta \neq 1$	On edge $\mathbf{v}_1^f\mathbf{v}_2^f$
$\alpha = 0$ and $\beta \neq 0$	On edge $\mathbf{v}_2^f\mathbf{v}_3^f$
$\beta = 0$ and $\alpha \neq 1$	On edge $\mathbf{v}_3^f\mathbf{v}_1^f$

3.2 Blending on surface

As the purpose of blending is to progressively move the points, $\mathbf{s}_j \in C_i$, towards the points, $\mathbf{s}_j^r \in C_{i+1}$, an ideal blending can be obtained by 1) computing a geodesic curve between $\mathbf{s}_j$ and $\mathbf{s}_j^r$ on the working surface M and 2) searching a point $\hat{\mathbf{s}}_j$ on the geodesic curve such that geodesic distance from $\hat{\mathbf{s}}_j$ to $\mathbf{s}_j$ is t of the geodesic curve's length (where $t = j/n$ is used in our spiraling algorithm). However, computing an exact geodesic curve on a piecewise linear surface is time-consuming (ref. [20]) and the approximate solution can lead to non-smooth result (see Fig.5(a)). We develop an algorithm below to achieve a good tradeoff between the speed and the accuracy.

The basic idea of our method is to drag the point $\mathbf{s}_j$ along the working surface M to a place which has the BDM value $(i+\frac{j}{n})\omega$. The point $\mathbf{s}_j^r$ serves as the dragger during this movement. Specifically, the dragging algorithm is developed using barycentric coordinates. For a current position, $\mathbf{s}_j^c$, of the point $\mathbf{s}_j$, a triangle face f holding the point $\mathbf{s}_j^c$ is always kept during the dragging. Our algorithm focuses on how to move the point inside f to a new position.

First of all, the barycentric coordinate (α,β) of $\mathbf{s}_j^c$ in f is computed so that

$$\mathbf{s}_j^c = \mathbf{f}(\alpha,\beta) = \alpha\mathbf{v}_1^f + \beta\mathbf{v}_2^f + (1-\alpha-\beta)\mathbf{v}_3^f$$

with $\mathbf{v}_1^f$, $\mathbf{v}_2^f$ and $\mathbf{v}_3^f$ being the three vertices of f and $\mathbf{f}(\cdots)$ returning the position of a point in f determined by barycentric coordinate. Whether the point $\mathbf{s}_j^c$ is located at the boundary of f can be determined by the conditions listed in Table 2. Note that, in our discrete BDM computation (Definition 4), the field value at an arbitrary point on the surface is also computed by using barycentric coordinate and the field values stored at the vertices, that is

$$\bar{d}_g^B(\mathbf{s}_j^c) = \alpha\bar{d}_g^B(\mathbf{v}_1^f) + \beta\bar{d}_g^B(\mathbf{v}_2^f) + (1-\alpha-\beta)\bar{d}_g^B(\mathbf{v}_3^f).$$

By this, it is easy to prove the following proposition.

Proposition 1 For three points, $\mathbf{q}$, $\mathbf{p}_a$ and $\mathbf{p}_b$, on the same triangle f, if $\mathbf{q} = (1-t)\mathbf{p}_a + t\mathbf{p}_b$, their BDM values satisfy

$$\bar{d}_g^B(\mathbf{q}) = (1-t)\bar{d}_g^B(\mathbf{p}_a) + t\bar{d}_g^B(\mathbf{p}_b). \tag{3}$$

Proof. For

$$\mathbf{p}_a = \alpha_a \mathbf{v}_1^f + \beta_a \mathbf{v}_2^f + (1-\alpha_a-\beta_a)\mathbf{v}_3^f,$$
$$\mathbf{p}_b = \alpha_b \mathbf{v}_1^f + \beta_b \mathbf{v}_2^f + (1-\alpha_b-\beta_b)\mathbf{v}_3^f,$$

we also have

$$\bar{d}_g^B(\mathbf{p}_a) = \alpha_a \bar{d}_g^B(\mathbf{v}_1^f) + \beta_a \bar{d}_g^B(\mathbf{v}_2^f) + (1-\alpha_a-\beta_a)\bar{d}_g^B(\mathbf{v}_3^f),$$
$$\bar{d}_g^B(\mathbf{p}_b) = \alpha_b \bar{d}_g^B(\mathbf{v}_1^f) + \beta_b \bar{d}_g^B(\mathbf{v}_2^f) + (1-\alpha_b-\beta_b)\bar{d}_g^B(\mathbf{v}_3^f).$$

When $\mathbf{q} = (1-t)\mathbf{p}_a + t\mathbf{p}_b$, by the definition of barycentric coordinate, we can obtain

$$\mathbf{q} = \alpha \mathbf{v}_1^f + \beta \mathbf{v}_2^f + (1-\alpha-\beta)\mathbf{v}_3^f$$

with

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = (1-t)\begin{pmatrix} \alpha_a \\ \beta_a \end{pmatrix} + t\begin{pmatrix} \alpha_b \\ \beta_b \end{pmatrix}. \tag{4}$$

By Definition 4,

$$\bar{d}_g^B(\mathbf{q}) = \alpha \bar{d}_g^B(\mathbf{v}_1^f) + \beta \bar{d}_g^B(\mathbf{v}_2^f) + (1-\alpha-\beta)\bar{d}_g^B(\mathbf{v}_3^f). \tag{5}$$

Substituting Eq.(4) into Eq.(5),

$$\begin{aligned} \bar{d}_g^B(\mathbf{q}) &= ((1-t)\alpha_a + t\alpha_b)\bar{d}_g^B(\mathbf{v}_1^f) \\ &\quad +((1-t)\beta_a + t\beta_b)\bar{d}_g^B(\mathbf{v}_2^f) \\ &\quad +((1-t) + t - ((1-t)\alpha_a + t\alpha_b) \\ &\quad -((1-t)\beta_a + t\beta_b))\bar{d}_g^B(\mathbf{v}_3^f) \\ &= (1-t)\bar{d}_g^B(\mathbf{p}_a) + t\bar{d}_g^B(\mathbf{p}_b). \end{aligned}$$

□

This proposition will be used to derive the terminal condition of dragging below.

Secondly, a plane P is formed by the vector $\mathbf{s}_j^c\mathbf{s}_j^r$ and f's normal vector $\mathbf{n}_f$, where P passes through the point $\mathbf{s}_j^c$ and has the normal vector as $\mathbf{s}_j^c\mathbf{s}_j^r \times \mathbf{n}_f$. The intersections between P and the edges of f are computed. Two configurations of intersections can be found.

- When the point $\mathbf{s}_j^c$ is located on an edge e, the intersection *not* on e is used as the *ghost* for dragging.
- When $\mathbf{s}_j^c$ is not on the boundary of f, the intersections are located at different sides of $\mathbf{s}_j^c$. Among them, the one located at the same side of $\mathbf{s}_j^r$ is selected (see Fig.6 for an illustration).

The barycentric coordinate of the ghost, (α_g, β_g), is also computed.

Now we can search for an optimal point that gives the BDM value as $(i + \frac{j}{n})\omega$ along the line between $\mathbf{s}_j^c$ and its ghost on the face f. The search is conducted in barycentric coordinates. Specifically, we are looking for a point with the barycentric coordinate

$$(\alpha_{opt}, \beta_{opt}) = ((1-t)\alpha + t\alpha_g, (1-t)\beta + t\beta_g), \tag{6}$$

which has

$$\bar{d}_g^B\left(\mathbf{f}(\alpha_{opt}, \beta_{opt})\right) = (i + \frac{j}{n})\omega. \tag{7}$$

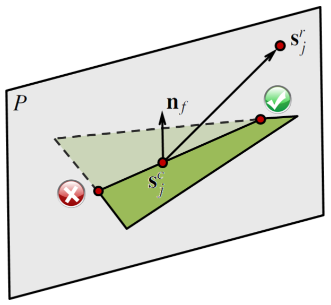


Figure 6: An illustration of the dragging taken inside a triangle.

According to the property introduced in Proposition 1 and Eq.(7), the value of t can be determined by

$$(1-t)\bar{d}_g^B(\mathbf{s}_j^c) + t\bar{d}_g^B(\mathbf{f}(\alpha_g, \beta_g))) = (i + \frac{j}{n})\omega. \tag{8}$$

When $t < 1$, the optimal position of moving $\mathbf{s}_j^c$ onto the spiral curve has been found as $\mathbf{f}(\alpha_{opt}, \beta_{opt})$ with $(\alpha_{opt}, \beta_{opt})$ determined by Eq.(6). When $t > 1$, it means that the optimal point cannot be found in the triangle f. The edge, e_g, that provides the ghost point for dragging in the triangle f will be used to determine the next triangle to be examined in the next step of dragging. Specifically, the other face adjacent to e_g will be used to search for the point which gives the target BDM value $(i + \frac{j}{n})\omega$. The traversal on the connected triangles is repeated until the optimal point is found. An example result can be found in the right of Fig.5.

4 BDM-Based Decomposition

Iso-contours of the BDM are generated on the working surface, M, to analyze whether M can be covered an intersection-free spiral curve with nearly even distances – the distance between neighboring spirals is expected to be a constant. If this cannot be satisfied, M must be decomposed into smaller regions to be covered by spiral curves. This section presents the methods for 1) analyzing iso-contours and 2) decomposing M by the topology information of iso-contours.

4.1 Topological analysis by iso-contours

Starting from the boundary, the iso-curves with constant values of BDM can be generated on the working surface M one by one. Specifically, for generating the i-th iso-contour with the value of BDM as $\bar{d}_g^B(\mathbf{p}) \equiv i\omega$ with $i \in \mathbb{Z}$, the BDM values on every vertices are compared with the iso-value (i.e., $i\omega$). For a triangle f with its vertices' BDM-values both smaller and equal/greater than $i\omega$, the iso-curve in this triangle can be approximated by a straight-line. The intersection between the iso-curve and a triangle edge e is first determined by linear interpolation when the BDM-values of a vertex on e is $< i\omega$ and the other vertex's BDM-value is $\geq i\omega$. Two such intersections can be found in f and be connected by a line segment. Linking such line segments forms a piecewise lin-

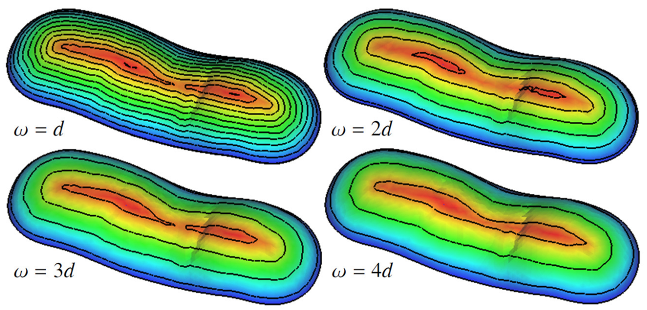


Figure 7: Increasing the values of ω from d to $4d$, different conclusion can be made by the topology analysis on the same surface path. Here d is the average edge length of the input mesh M.

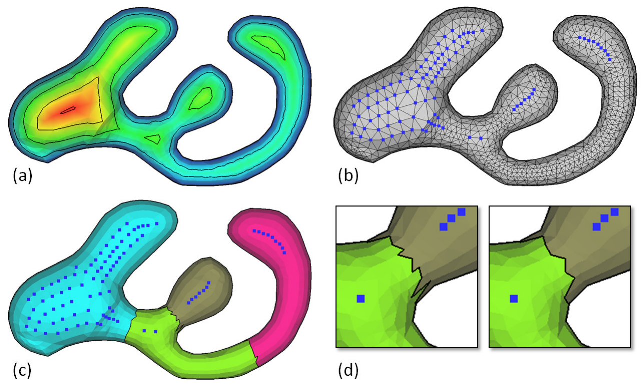


Figure 8: BDM guided surface decomposition: (a) iso-contours with complex topology, (b) the seeds (displayed in blue dots) for segmentations, (c) discrete voronoi diagram of the seeds as an initial decomposition of the given surface, and (d) decomposed regions with smoothed boundaries. Regions as the result of decomposition are displayed in different colors.

ear approximation of the i-th iso-contour. See Fig.7 for an example.

Now we will analyze the topology of iso-contours on the BDM of a surface to see if it needs to be decomposed. As shown in section 3, our spiral curve generation algorithm requires the input surface having an ω-simple topology with $\omega = W$, where W is the distance between neighboring channels determined by the physical model. Here, we have to analyze the topology of a working surface, M. When its topology is not ω-simple, decomposition must be applied. Note that, this topological analysis is resolution-dependent – i.e., using different values for ω may lead to different conclusions. An illustration is given in Fig.7. Considering d as the average edge length of the input mesh M, the surface is *not* ω-simple when $\omega = d, 2d, 3d$. But it is ω-simple when $\omega = 4d$.

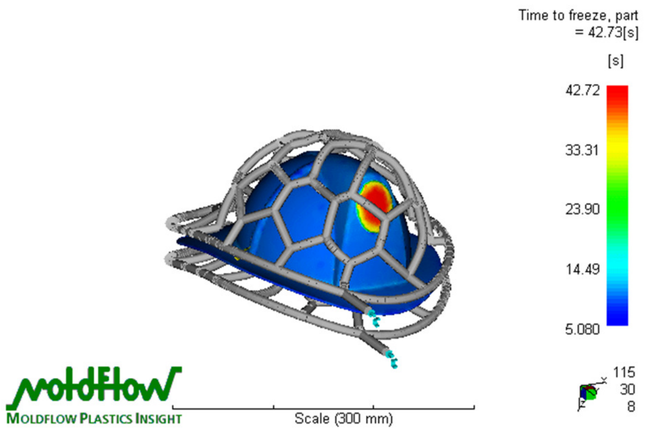
By linking the straight-lines inside triangles of M according to the iso-value $\equiv i\omega$, loops for the i-th iso-curve can be formed. In practice, since we only need to detect if the topology of an iso-curve is ω-simple, the implementation of analysis can be simplified as follows. Each line segment of an iso-curve has two endpoints with each located on an edge of M. Starting from one endpoint, we can travel along the line segments one by one with the help of the connectivity information of M. The travel ends by coming back to the starting point. If there is any segment belonging to the iso-curve but not be visited during the travel, the iso-curve should have multiple loops. In other words, the surface patch is not ω-simple and must be decomposed.

4.2 Decomposition algorithm

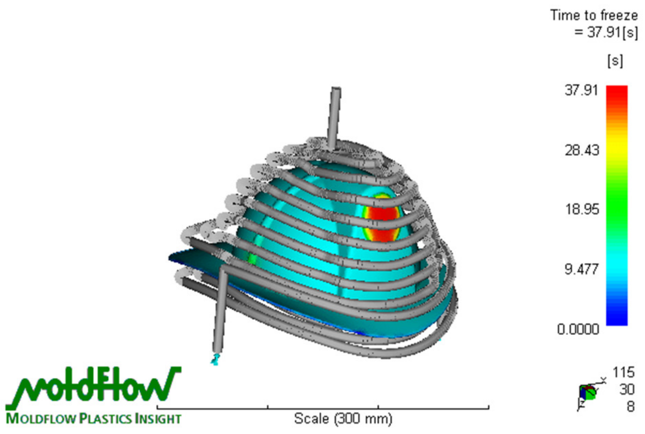
The decomposition of an input mesh surface M is governed by the BDM analysis as BDM already provides cues for decomposition. For example, when looking at the contours displayed in Fig.8(a), most people will subjectively explain that the surface is composed of four regions. The problem is how to extract this information and employ it in the decomposition.

We develop an automatic decomposition algorithm consisting of the following steps:

- First, starting from $i = 1$, the topology of iso-contour with $\bar{d}_g^B(\mathbf{p}) \equiv i\omega$ is checked one by one. When the i-th iso-contour has multiple loops, all the vertices with their BDM value greater than $i\omega$ will be specified as candidate seeds for region classification (see the blue dots in Fig.8(b)).
- Second, the seeds are grouped in clusters by a flooding algorithm. Basically, two seeds should be grouped into the same cluster if they belong to one edge and their BDM values are greater than $i\omega$. Take Fig.8(b) as an example, the seeds are classified into four clusters. The number of clusters determines the number of regions should be decomposed from input surface M.
- Third, the dual graph G of M is constructed, where each face of M is converted into a node of G and every two adjacent faces are linked by an edge. The weight on an edge in G is assigned as the distance between centers of faces that correspond to the two graph nodes linked by the edge. For each triangle adjacent to a seed determined in above steps, the triangle's corresponding node in G will be served as a source node. Triangles adjacent to the seeds in the same cluster will be assigned with the same ID and serve as the same source. Applying Dijkstra's algorithm with multiple sources on this graph G assigns each graph node an ID, which indicates its closest source on the weighted graph. As a result, every triangle in M is assigned an ID – i.e., M has been decomposed into regions consisting of triangles having the same ID. Triangles with the same ID are displayed in the same color in Fig.8(c).
- At last, the boundary between two regions, Ω_i and Ω_j, is smoothed by assigning a triangle of Ω_i into the region Ω_j if such an assignment can reduce the length of region boundary; similar processing is also applied to the triangles in Ω_j near the boundary. An example result can be found in Fig.8(d).



(a) VD-based cooling channel



(b) Spiral channels generated by this approach

Figure 10: Color maps for displaying the time of plastic part freezing to the ejection temperature by using (a) the VD-based cooling channel [5] vs. (b) the spiral channel generated by this approach.

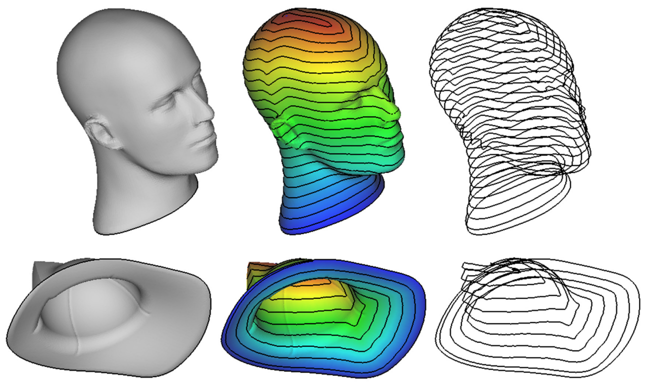


Figure 9: Spiral curves can be generated on models with dense mesh: (top) a head model with 31k triangles and (bottom) a helmet model with 53k triangles.

The result of decomposition should be verified by taking another round of BDM-based topological analysis in each region. If a region is not ω-simple, the algorithm above needs to be taken again to further decompose this region. This cycle of decomposition and verification must be repeatedly applied until all regions are ω-simple.

5 Results and Discussion

The algorithms presented in this paper are implemented as a program in C++. The spiral curves can be generated efficiently on models represented by two-manifold triangular meshes. For example, spiral curves can be generated for the models shown in Fig.9 with 31k triangles (the head model) and 53k triangles (the helmet model) in 344ms and 314ms respectively on a PC with Intel Core 2 Quad CPU Q6600 2.4GHz. Moreover, the BDM-based decomposition can also be computed efficiently. The spiral curves can be generated on all models shown in this paper in less than 2 seconds.

To verify the physical performance of cooling channels generated by our approach, simulations are taken on the injection molding simulation software – MoldFlow Insight [23]. Our program provides a function to write the piecewise spiral curves into a macro file, which can be imported into MoldFlow as the axial curves to generate the cooling channels automatically. Polypropylene (PP) is employed as the part material and steel P20 is used for the part-forming components of molds. Detailed parameters applied in our tests are listed in Table 3 and 4. Water is selected as the coolant and its temperature is assigned as 25°C. Two models are selected for the case study below. To demonstrate the benefit of conformal cooling channels generated on the offset surfaces, two thin-shell models – helmet and cell-phone cover – are selected. According to our experiences, the conformal cooling is most effective on such thin-shell models.

Table 3: Material Properties of Part and Mold

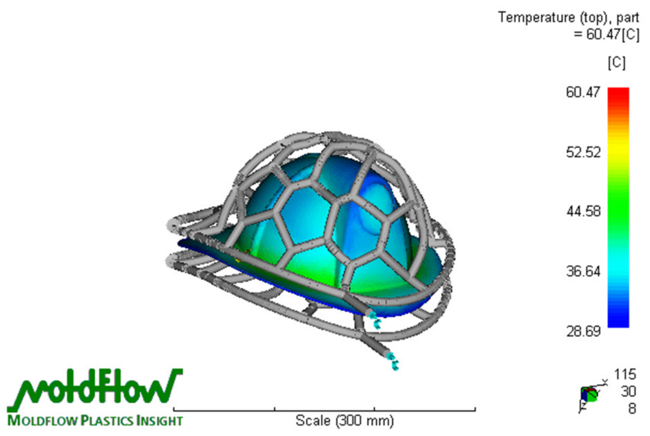
	P20	PP	ABS
Density $[kg/m^3]$	7800	900	1045
Specific heat $[J/(kg \cdot K)]$	460	1900	1950
Thermal conductivity $[W/(m \cdot K)]$	29	–	–
T_{melt} [°C]	–	220	230
T_{eject} [°C]	–	70	60

Table 4: Geometry Properties of Parts

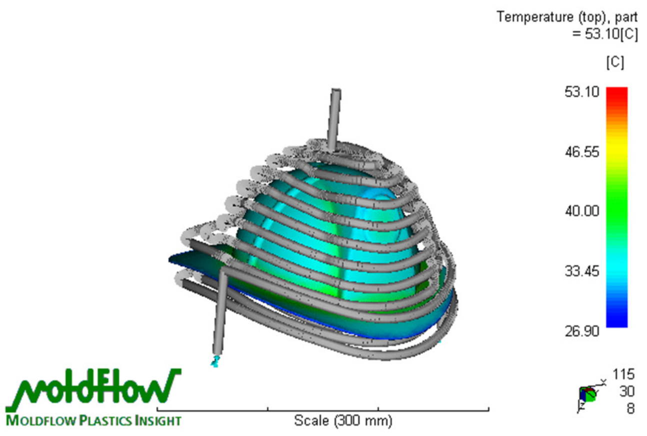
	Helmet	Cell Phone
Dimension $[mm]$	$[229.8, 316.6, 157.0]$	$[219.4, 45.76, 69.68]$
Thickness $[mm]$	2.500	2.000
Volume $[mm^3]$	2.540×10^5	4.946×10^4
Area $[mm^2]$	2.008×10^5	5.723×10^4

5.1 Case study I: helmet

The first case study is about the molding of a toy helmet made by ABS. According to industrial experience, the cy-

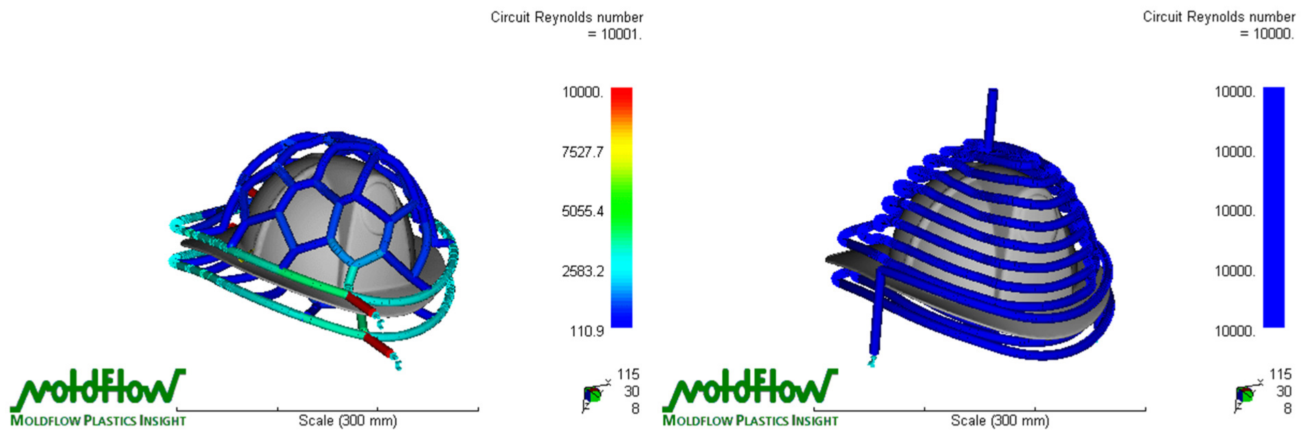


(a) VD-based cooling channel [5]

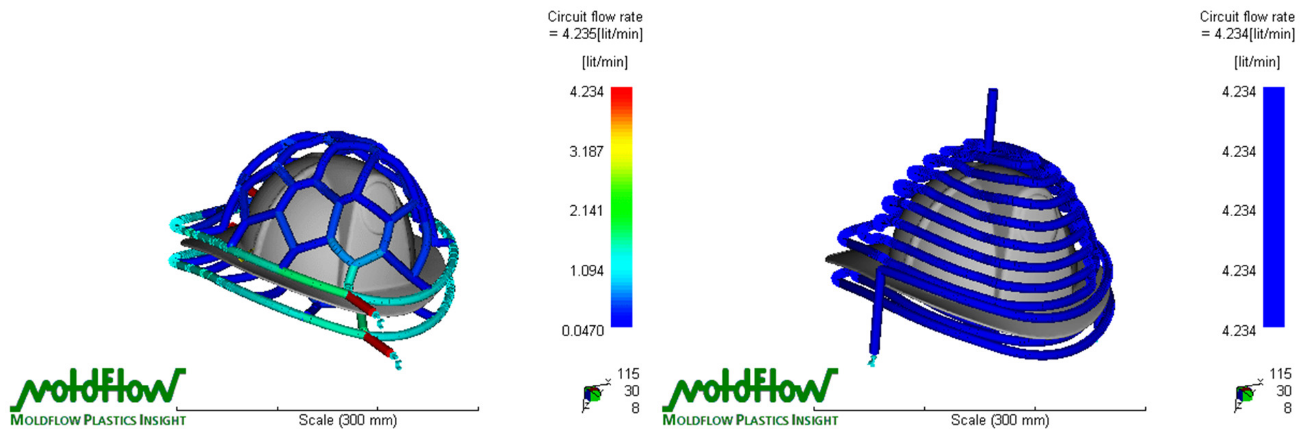


(b) Spiral channels generated by this approach

Figure 11: Color maps for displaying the average temperature on the mold surfaces during the cooling cycle.



(a) Comparison on Reynolds number



(b) Comparison on flow rate

Figure 12: Color maps for displaying the Reynolds number and flow rate in the VD-based and the spiral cooling channels. No variation can be found in the simulation result on the spiral channel generated by our method.

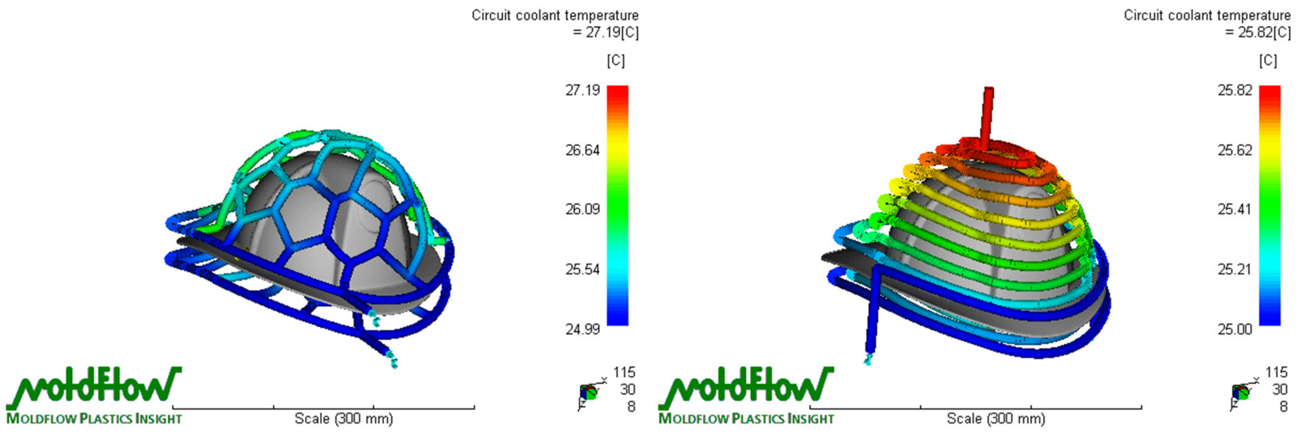


Figure 13: Color maps for displaying the temperature of coolant in the VD-based circuit vs. the spiral cooling channels.

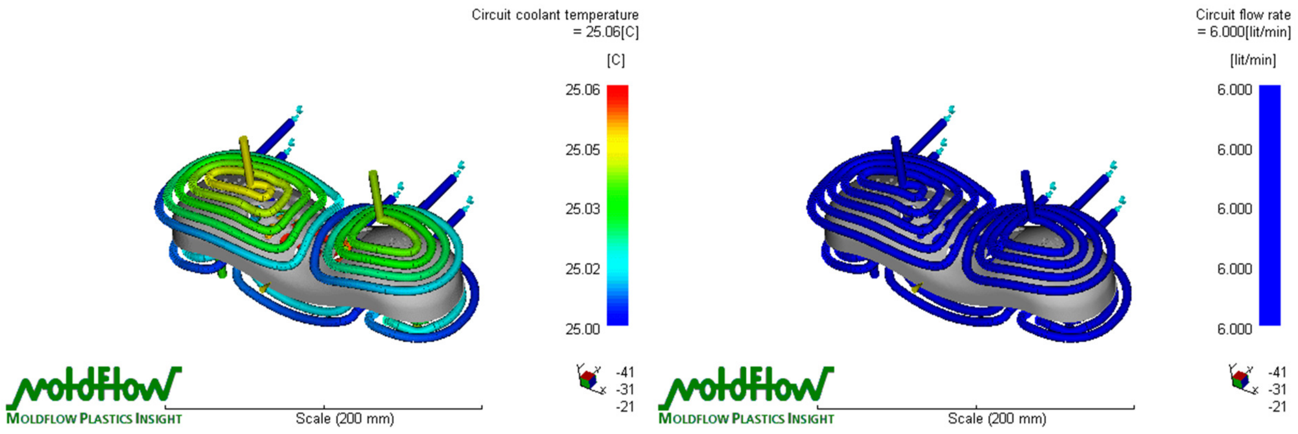


Figure 14: Color maps for displaying the temperature and the flow rate of coolant in a cooling system with multiple spiral channels. No variation can be found in the simulation results on the channels generated by our method.

cle time $t_{cycle} = 20s$ and the channel diameter $D = 10mm$ is selected. By the physical formulation of conformal cooling, the distance between cooling line to mold surface (l_m) and the cooling channel line pitch distance (W) is determined as $l_m = 12.7mm$ and $W = 20mm$. The cooling efficiency of the spiral cooling channels determined by this algorithm is compared with the *Voronoi Diagram* (VD) based conformal cooling circuits generated by our prior work [5], in which the cooling efficiency of VD-based channel has proved to be much higher than conventional straight channels. Both simulations use the same coolant – water at 25°C and with Reynolds number $Re = 10^5$.

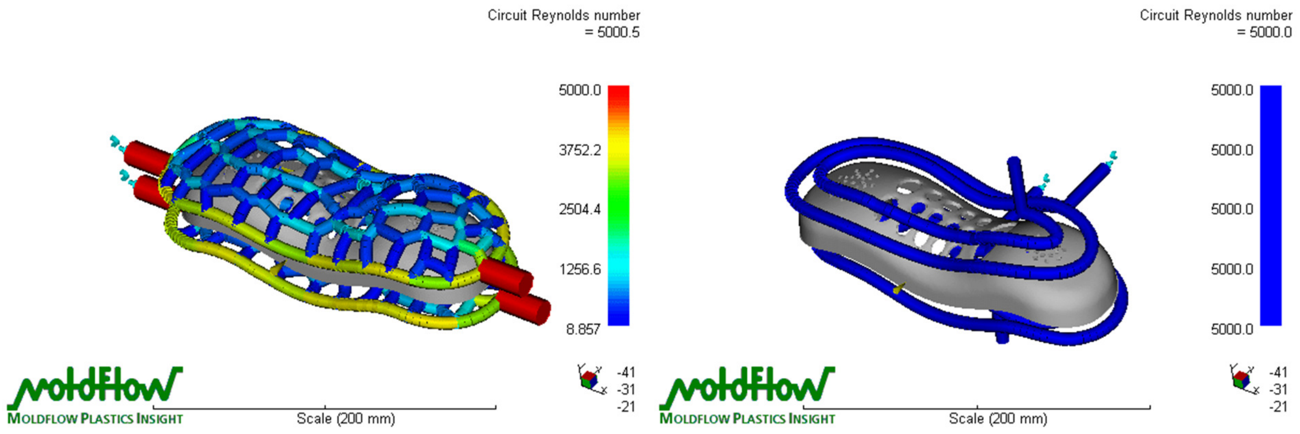
The simulation results are shown in Figs.10 and 11. It is easy to find that the cooling time (time to freeze) has been shortened from 42.73 sec. to 37.91 sec. The cooling also occurs more uniformly. The temperature variation between 28.69°C and 60.47°C has been reduced to the range between 26.90°C and 53.10°C. The obvious improvement in cooling efficiency is shown in Figs.10 and Fig.11. Fig.12 gives an explanation for this enhancement. It is clear that VD-based cooling circuit leads to poor flow rates and Reynolds numbers resulting in non-uniform cooling in the tool. The conformal spiral channel, by contrast, keeps a stable turbulent flow rate through the entire length and enables the heat to be transferred more effectively. As shown in Fig.13, the coolant temperature has a range of 2.2°C in the VD-based channel but the temperature rise of the coolant in our spiral channel is cut to 0.82°C. This demonstrates the increase of the cooling efficiency in spiral channels, and thus more uniform cooling.

5.2 Case study II: cell-phone cover

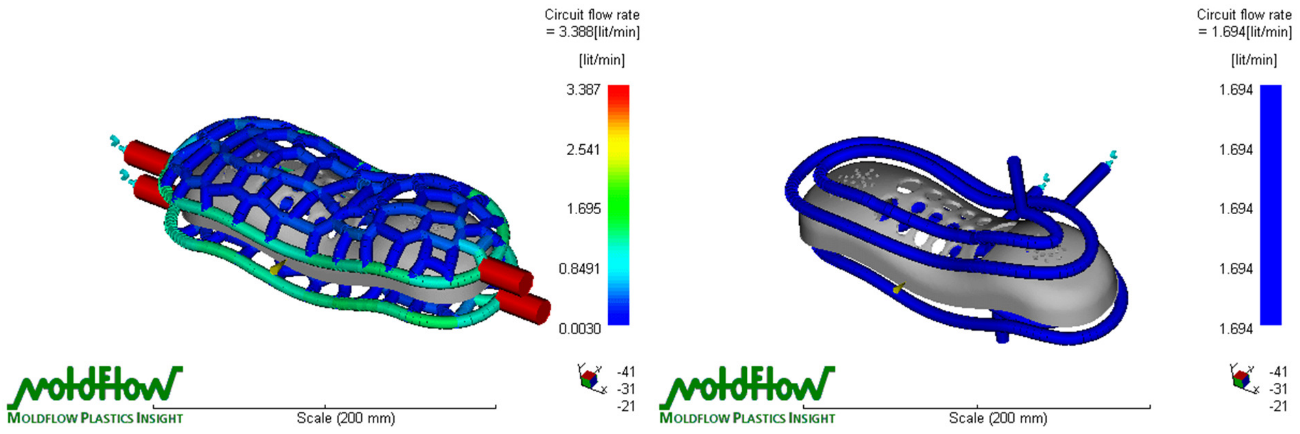
A cell-phone cover to be fabricated by polypropylene (PP) is tested in the simulation. Two spiral channels with different parameters are studied.

Firstly, a circuit with $D = 6mm$, $W = 10mm$ and $l_m = 11.0mm$ is generated. As shown in Fig.14, two spiral channels are constructed to cover the working surface in the upper mold and three channels are generated in the lower mold. Set inlet coolant flow rate to $6lit/min$, as an example. The coolant can maintain a nearly constant flow rate and the temperature rise is minimized to be within 0.06°C.

In the second study, the spiral channels are generated and compared with $D = 8mm$, $W = 21.4mm$ and $l_m = 11.0mm$, which is sparser than the above case. For comparison purpose, VD-based cooling circuits are generated with the same set of parameters. To achieve the same Reynolds number at



(a) Comparison on Reynolds number



(b) Comparison on flow rate

Figure 15: Color maps for displaying the Reynolds number and flow rate in the VD-based and the spiral cooling channels. No variation can be found in the simulation result on the spiral channel generated by our method.

the inlet as $R_e = 5,000$, the pumping expenses drastically increase for the VD-based circuit. As shown in Fig.15, a flow rate with $3.388 lit/min$ is required at the inlet of VD-based circuit but the spiral channel needs only about $1.694 lit/min$ at its inlet. Moreover, for the VD-based circuit, the Reynolds number drops significantly (the flow is no longer turbulent) in the branches of channels even after supplying a high pumping pressure at the inlet. This leads to an inefficient heat transfer. Again, comparisons on the temperature of coolant are shown in Fig.16, where the temperature differences of the coolant are 2.32°C for the VD-based circuit and only 0.39°C for the spiral channels.

5.3 Limitations

Though the proposed method works well on the two models shown in the case study, it may have problems on parts with more complex topology or larger variations on the thickness. For example, when the working surface of a part generated by offsetting has multiple holes, some manual defeaturing operations are needed to simplify the topology of the part and also the working surface. In terms of heat transfer in the injection mold, a constant offset conformal cooling channel does not always ensure an even cooling due to corner effects and thickness variations. Our current approach may not give very good performance under these scenarios.

Constraints in mold design are not considered when generating the spiral cooling channels. For example, injection molds have ejector pins used to separate the finished parts from the mold itself. At the places where the ejector pins are planned to install, the working surface must be trimmed off to ensure no intersection between the pins and the cooling channels. Considering such constraints will lead to a more difficult task of designing cooling channels. Furthermore, the current model is developed for the mold with two parts – upper/lower molds. When applying it to complex molds made of several moving parts, more constraints need to be considered.

6 Conclusion

We present an approach in this paper to generate spiral and conformal cooling channels for plastic injection molding of parts with high-curved surfaces. This approach shows advantages in two aspects:

- First, the cooling channels are generated on a working surface that is offset from the cavity surface of a mold.

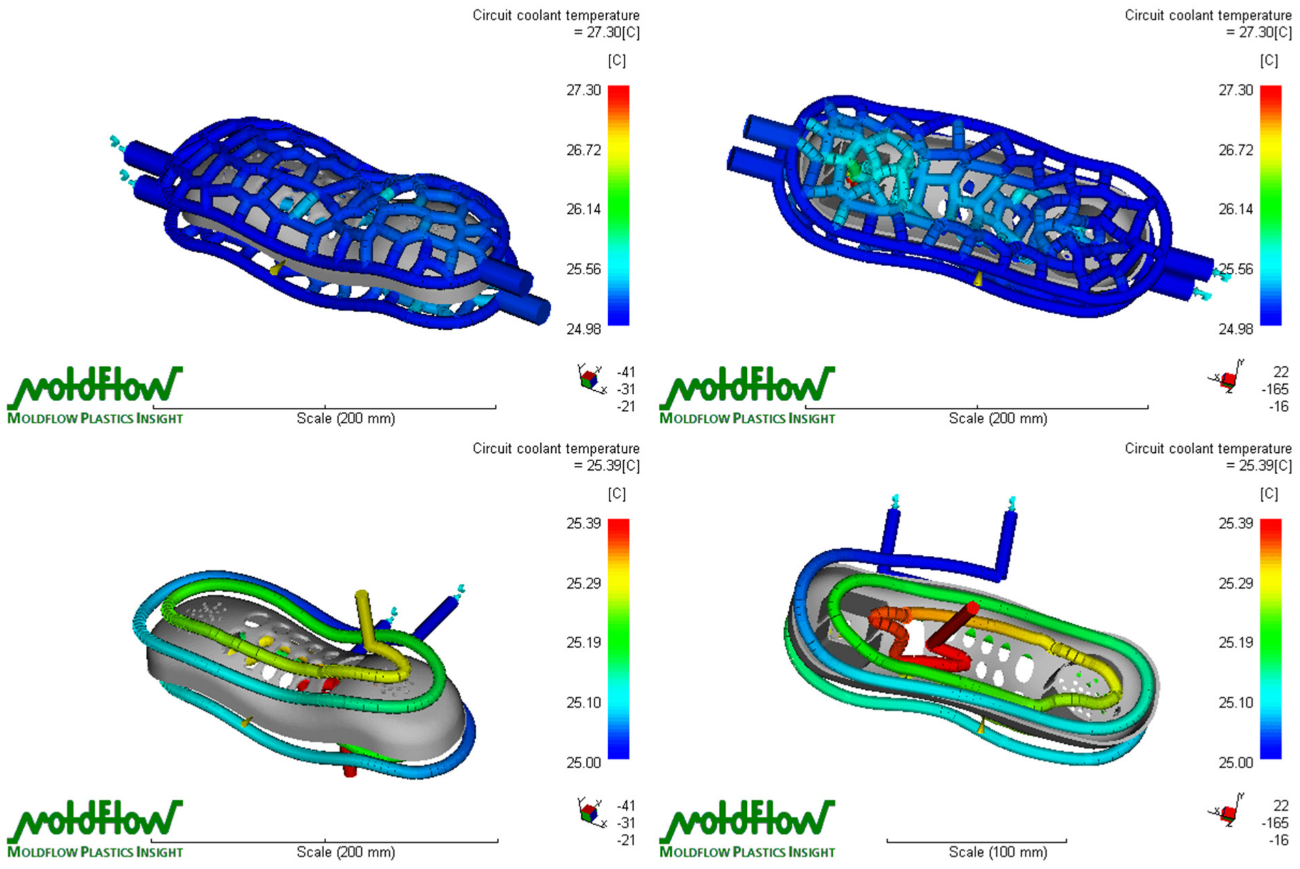


Figure 16: Comparison on the temperature distribution in the VD-based cooling circuits (left) and the spiral cooling channels (right).

As a result, conformal cooling can be obtained. This is a characteristic that holds for our prior work on conformal cooling circuits [5] but not for the conventional cooling channels [1].

- Second, the spiral cooling channels are generated in this approach so that the flow of coolant can be kept in a high speed in the channels. This is a characteristic that holds for conventional cooling channels but not for conformal cooling circuits.

In summary, this new approach can generate conformal cooling channels with higher flow rate so that heat transfer efficiency is improved in plastic injection molding. Moreover, comparing to our prior work on conformal cooling circuits, the channels generated by this new approach are easier to fabricate by using copper duct bending instead of expensive selective laser sintering.

In order to realize the approach for generating spiral and conformal cooling channels, the technical contributions introduced by this work include an efficient algorithm for smooth spiraling and a BDM analysis based decomposition algorithm. With the help of these two algorithms, smooth spiral curves can be efficiently generated on the working surface to serve as the axial curves of cooling channels.

A possible future work is to incorporate the offset surface generation algorithm into the BDM computation and the curve spiraling. We plan to generate directional offset surface patches in the form of point-sampled surface by using the distance function defined in [12]. Then, the BDM can be computed on a graph generated on the point set, which defines a *moving least-square* (MLS) surface for generating the spiral curves. Another practical future work is to test mold performance with spiral and conformal cooling channels in the industrial production. This will further verify the advantages of spiral and conformal cooling channels.

Acknowledgments

The authors would like to thank the staff of the Digital Factory at the Hong Kong Polytechnic University for their technical support. The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong SAR, China (Project No.: PolyU 5368/09E).

References

[1] E. Sachs, E. Wylonis, S. Allen, M. Cima, H. Guo, Production of injection molding tooling with conformal cooling channels using the three dimensional printing process, Polymer Engineering and Science 40 (5) (2000) 1232–1247.

[2] X. Xu, E. Sachs, S. Allen, The design of conformal cooling channels in injection molding tooling, Polymer Engineering and Science 41 (7) (2001) 1265–1279.

[3] H. S. Park, N. H. Pham, Design of conformal cooling channels for an automotive part, International Journal of Automotive Technology 10 (1) (2009) 87–93.

[4] C. L. Li, A feature-based approach to injection mould cooling system design, Computer-Aided Design 33 (14) (2001) 1073–1090.

[5] Y. Wang, K.-M. Yu, C. C. L. Wang, Y. Zhang, Automatic design of conformal cooling circuit for rapid tooling, Computer-Aided Design 43 (8) (2011) 1001–1010.

[6] Y. F. Sun, K. S. Lee, A. Y. C. Nee, The application of u-shape milled grooves for cooling of injection moulds, IMechE Proceedings Part B, Journal for Engineering Manufacture 216 (2002) 1561–1573.

[7] J. C. Ferreira, A. Mateus, Studies of rapid soft tooling with conformal cooling channels for plastic injection moulding, Journal of Material Processing Technology 142 (2003) 508–516.

[8] A. B. M. Saifullah, S. H. Masood, Finite element thermal analysis of conformal cooling channels in injection moulding, in: Australasian Congress on Applied Mechanics, Brisbane, Australia, 2007.

[9] J. M. Jauregui-Becker, G. Tosello, F. J. van Houtena, H. N. Hansenb, Performance evaluation of a software engineering tool for automated design of cooling systems in injection moulding, Procedia CIRP 7 (2013) 270–275.

[10] J. R. Rossignac, A. A. G. Requicha, Offsetting operations in solid modelling, Computer Aided Geometric Design 3 (2) (1986) 129–148.

[11] S. Liu, C. C. L. Wang, Fast intersection-free offset surface generation from freeform models with triangular meshes, IEEE Transactions on Automation Science and Engineering 8 (2) (2011) 347–360.

[12] C. C. L. Wang, Y. Chen, Thickening freeform surfaces for solid fabrication, Rapid Prototyping Journal 19 (6) (2013) 395–406.

[13] M. B. Bieterman, D. R. Sandstrom, A curvilinear tool-path method for pocket machining, ASME Journal of Manufacturing Science and Engineering 125 (4) (2003) 709–715.

[14] Z. Yao, A. Joneja, A path generation for high speed machining using spiral curves, Computer-Aided Design and Applications 4 (1) (2007) 191–198.

[15] M. Held, C. Spielberger, A smooth spiral tool path for high speed machining of 2D pockets, Computer-Aided Design 41 (7) (2009) 539–550.

[16] Q. Zou, J. Zhao, Iso-parametric tool-path planning for point clouds, Computer-Aided Design 45 (11) (2013) 1459–1468.

[17] M. P. Do-Carmo, Differential Geometry of Curves and Surfaces, Prentice Hall, 1976.

[18] Y.-J. Liu, Exact geodesic metric in 2-manifold triangle meshes using edge-based data structures, Computer-Aided Design 45 (3) (2013) 695–704.

[19] Y.-J. Liu, Z. Chen, K. Tang, Construction of iso-contours, bisectors, and voronoi diagrams on triangulated surfaces, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1502–1517.

[20] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. Gortler, H. Hoppe, Fast exact and approximate geodesics on meshes, ACM Transactions on Graphics 24 (3) (2005) 553–560.

[21] J. Chen, Y. Han, Shortest paths on a polyhedron, in: Proceedings of the sixth annual symposium on Computational geometry, 1990, pp. 360–369.

[22] T. Kanai, H. Suzuki, Approximate shortest path on a polyhedral surface and its applications, Computer Aided Design 33 (11) (2001) 801–811.

[23] Autodesk, MoldFlow Insight, http://www.autodesk.com, 2010.