Abstract

In this thesis, we study the so-called distance model for analyzing the filling stage in injection moulding of plastics. This is done in the case of non-uniform thickness of the mould and planar as well as non-planar objects with thin walls. We are lead to consider the so-called weighted region problem, i.e. a problem of shortest path. We solve it by using Dijkstra's algorithm for computing shortest path in an associated graph.

We give a brief background to the problem of injection moulding and the derivation of the distance model. Results from numerical simulations in comparison with experiments or other simulations found in the literature are given.

AMS Subject Classification (1991). Primary: 49
Secondary: 76

Key words and phrases: injection moulding, Hele–Shaw flow, Dijkstra's algorithm, pseudo-circle, distance, window sequence, convexity.

Acknowledgement

This thesis is a part of a licentiate degree, and I want to express my sincerest gratitude to my supervisor Professor Gunnar Aronsson for introducing me into the field of applied mathematics, in particular the problem of injection moulding. Without his support and advice, both on the subject and the presentation of it, this thesis would not exist.

I also wish to express my gratitude to:

All friends and colleagues at the Department of Mathematics, for interesting and stimulating discussions on mathematics as well as non-mathematical topics.

Last but not least, my fiancée Malin. Without her encouragement and support I doubt I would have succeeded in finishing this thesis.

Linköping, February 1997
Peter Johansson

Rather than love, than money, than fame, give me truth.

Henry David Thoreau
(Walden, 1854)

I have approximate answers and possible beliefs and different degrees of certainty about different things, but I'm not absolutely sure of anything, and many things I don't know anything about, such as whether it means anything to ask why we're here, and what the question might mean. I might think about it a little bit, but if I can't figure it out, then I go on to something else. But I don't have to know an answer. I don't have to... I don't feel frightened by not knowing things, by being lost in the mysterious universe without having any purpose, which is the way it really is, as far as I can tell, possibly. It doesn't frighten me.

Richard Phillips Feynman (1918-1988)

Contents

0. Introduction

A polymeric material is a very versatile raw material as can be seen from the fact that the products, often called plastics, appear frequently in the daily life of an ordinary citizen in the industrialized world of today. It is not only the chemical, physical and electrical properties that render plastics so popular, it is also the plastic deformable state achieved by thermoplastic polymers at higher temperatures, and possessed by thermosetting polymers before being chemically "set", that allows them to be shaped into a myriad of products, some of them of great geometric complexity. The shaping operations used are relatively quick and easy, and are well suited for mass production. However, the increasing geometric complexity of the products combined with the demand for short times for design of the manufacturing tools (such as moulds) increases the need for computer assistance in the design phase.

Professor Gunnar Aronsson has since 1989 developed a research group concerned with mathematical modeling of some non-Newtonian flow, in particular polymer shaping technologies using asymptotic methods and e.g. convex analysis to obtain a simple and intuitive model for injection moulding. The benefit of the theoretical analysis is deeper understanding and the possibility to extend the model to obtain better accuracy. The work started with a Master's Thesis by Lennart Pettersson [43] 1989, followed by Master's Theses by Torbjörn Andersson [3] 1990 and Mats Aigner [2] 1992, continued by a Licentiate Thesis by Klas Samuelsson [46] and a Doctoral Thesis by Ulf Janfalk [26] 1993, and recently Master's Theses by Mikael Lundquist [33] 1995 and Fredrik Berntsson [9] 1996. Janfalk's thesis is written essentially in pure mathematics but the results obtained are of great value for our modeling of injection moulding *and* compression moulding (work in progress by Andréas Bergwall). This Licentiate Thesis finished in 1997 concludes the current state of the research on polymer shaping using injection moulding in Gunnar Aronsson's group.

Instead of using the standard approach of solving the flow equations using some finite element method we use the so-called distance model to obtain a simpler algorithm. Then, instead of being forced to write complex software and use powerful computers, we are able to present a simple algorithm which runs fast on an ordinary PC. The weaknesses of the simple method we use are easier to analyze and understand than e.g. the complex interaction of mesh generation and flow-front tracking that renders problems hard to understand and analyze in the standard approach. We can not claim better accuracy than the standard approach, but have a simpler program that performs the computations without user interaction and still is not depending on extensive heuristic selections of parameters critical for the numerical method. In particular, no mesh needs to be created.

The purpose of this thesis is to study the concept of the distance model in the case of non-uniform thickness of the mould and non-planar objects with thin walls. Currently we have restricted us to a domain decomposable into polygons with constant thickness to obtain an effective and simple algorithm but the method as such does not impose such restrictions. As in the case of uniform thickness we are using Dijkstra's algorithm to compute the shortest path in a graph (that is modified to cope with the case of non-uniform thickness) and this turns out to give an algorithm that is fast and simple to implement.

0.1. The Structure of the Thesis

In Section 1 we give some background on the physical process of injection moulding and the important concept of Hele–Shaw flow which is essentially justified by the fact that the walls are thin compared to the other dimensions. Further, some phenomena that are interesting to predict are described and a few comments on simulation software are given.

In Section 2 we describe the ideas and assumptions leading to the distance model; a summary of the physical background is given, as well as the most important equations leading to the pseudo-circle principle.

In Section 3 we summarize known results in the case of uniform thickness as developed by Gunnar Aronsson and programmed and investigated by Mats Aigner and Mikael Lundquist, as well as results by others in constructing and investigating the complexity of algorithms for solving the problems that appear.

In Section 4 we state and analyze the main problem of the investigation in this thesis, namely The Weighted Region Problem (WRP); we also summarize results by others. We give a proof of the existence of a shortest path in the WRP, point out some minor mistakes in published papers, investigate properties of a shortest path and give a uniqueness result under certain natural restrictions. By analyzing a specific case we illustrate some phenomena that can occur in the WRP.

In Section 5 we formulate our algorithm for solving the WRP stated in Section 4 as well as some investigations of the complexity of the algorithm.

In Section 6 we investigate an idea for improving the method, still using the formulation as a WRP, by including influence from the boundary of the mould.

In Section 7 we give the results of our numerical experiments, including execution time on different computers and some remarks regarding an error estimate given in Section 5.

In Section 8 we explain our visualization of results from our computations, specifically how to detect one interesting phenomenon described in Section 1.3, namely weld lines. The problem of detecting air traps from the results of our computations has been solved by Fredrik Berntsson in [9].

In Section 9 we make some final remarks on the method and make suggestions for future research.

1. Background

This thesis is written with one application in mind, namely the filling pattern during injection moulding of thermoplastic products. This section will put this application into a context.

1.1. Injection moulding of plastic products

One of the most commonly used methods for manufacturing plastic products is injection moulding. Other methods include extrusion, calendering, compression and transfer moulding, and thermoforming. Extrusion is well suited for wire coating, rods, tubes, pipes and so on. By combining extrusion with blowing, films can also be manufactured. A blow mould connected to an extruder is often used for manufacturing bottles. Calendering is well suited for films and sheets. Transfer moulding can be used for plastic products reinforced with fibers in the form of a woven mat. The number of articles manufactured by injection moulding is immense, from gears, boxes, and domestic tools to bumpers and garbage cans; they are literally everywhere in our daily life.

The main advantages of injection moulding are geometric flexibility and short production cycle. This often eliminates the need of assembly of parts or post-processing of the product and the process is well suited for large series.

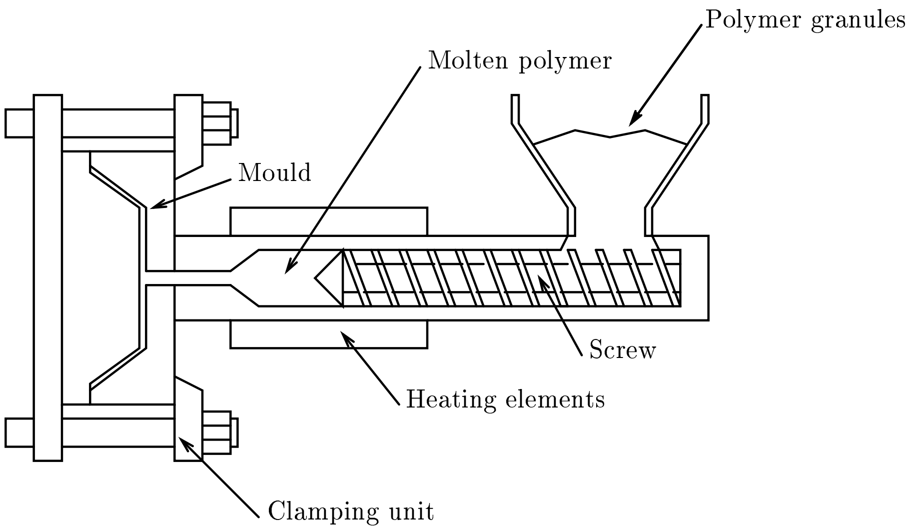


Figure 1: Schematic diagram of an injection moulding machine.

The injection moulding process consists of three major stages, filling, packing, and cooling. An injection moulding machine is schematically shown in Figure 1. A typical production cycle in injection moulding can be described as follows. The injection unit consist of a screw that is rotating inside a barrel. First, polymer granules are fed forward in the barrel while the screw retreats.

The granules is melted by the heat from heating elements and friction. When sufficient molten material for a "shot" is in front of the screw, it stops rotating and starts to move forward as a ram. As the screw moves forward it pushes the polymer melt into the mould through a nozzle; this is the *filling* or injection stage. When the mould is just filled the pressure is maintained for a certain period of time, called the *packing* stage. During the packing stage a little more material enters the mould to compensate for shrinking during cooling and solidification. When the polymer at the "gate", which is a narrow entrance to the mould, has frozen the *cooling* stage starts. During this stage the pressure drops as the melt cools and solidifies. When the material has solidified enough the mould opens and the product is ejected. During the cooling stage a new shot has been prepared as described above, the clamping unit closes and a new injection stage is started.

In this thesis we will only analyze the filling stage.

1.1.1. Types of polymers

There are two basic types of polymers, thermosets and thermoplastics. Thermosets become fluent by heating, but at some temperature a chemical process called cross-linking occurs and this solidifies the liquid into a rigid lattice structure. After cross-linking, the material will not remelt upon heating. Thermoplastics are also made fluent by heating, but solidify by cooling. This procedure can be repeated and thus thermoplastics can be recycled. The main difference in processing thermosets and thermoplastics is the mould temperature. For thermoplastics the mould is run "cold", and for thermosets it is "hot". We will focus on injection moulding of thermoplastics. Of the raw material used for injection moulding thermoplastics account for about 80%.

1.1.2. Modeling material response

For mathematically analyzing filling of the mould one needs equations governing the flow and a description of material response. The flow equations are obtained using principles of conservation of mass, momentum and energy. The material description, a so-called constitutive law, describes the material response to deformation. For some important classes of fluids it essentially relates rate of deformation to stress. If this relation is linear (at constant pressure and temperature) the fluid is called Newtonian. However, for polymer melts this is not the case; they exhibit a non-Newtonian behaviour. There are several constitutive laws, of varying complexity and accuracy, available describing polymer melts. A complex constitutive law does not only give rise to more complex equations to solve, it may also have many material parameters that could be hard to determine experimentally. Most constitutive laws used for flow analysis in injection moulding are of the generalized Newtonian type, i.e. of the type

$$\mathbf{T} = -p\mathbf{I} + 2\eta(\dot{\gamma}, T)\mathbf{D}, \tag{1.1}$$

where

$\mathbf{T}$	is the Cauchy stress tensor
p	is the pressure
T	the temperature
$\dot{\gamma}$	is the shear rate
$\mathbf{D}$	is the rate of deformation tensor

The rate of deformation tensor is the symmetric part of the velocity gradient and is related to the shear rate by $\dot{\gamma} = \sqrt{2\mathbf{D} : \mathbf{D}}$, where : denotes the standard inner product on the space of tensors.

The function η is called the apparent viscosity and is a characteristic of the material in question. Note that the generalized Newtonian model does not take any viscoelastic effects into account; this is however considered reasonable for analyzing the injection mould filling process and is used in many commercial packages for simulating the process. Simulations with those packages provide reasonable results in a number of relevant cases. One simple constitutive law of generalized Newtonian type is the so-called power-law model which states that

$$\eta(\dot{\gamma}, T) = K\dot{\gamma}^{n-1}, \tag{1.2}$$

where K is called the consistency and n the power-law index. For the Newtonian case we have $n = 1$ and $K = \mu =$ the viscosity. A typical value for the power-law index of polymer melts is $n = 0.3$ and we certainly have $0 < n < 1$. The behaviour that the (apparent) viscosity decreases with increasing rate of deformation is called shear-thinning or pseudoplasticity. The power-law model fits measured viscosity data reasonably well for most polymer melts if we restrict the shear rate to some interval called the power-law region. The main deficiency is that for low shear rates (outside the power-law region) the measured viscosity approaches a constant value called the Newtonian plateau and this is not predicted by (1.2). Other generalized Newtonian models try to take this effect into account in some way. However, there is some evidence that the power-law index has little effect on the filling pattern.

1.1.3. A brief note on the evolution of the injection moulding process and simulations

John Wesley Hyatt, a pioneering figure in the field of polymers, obtained a patent on a piston driven injection moulding machine for polymers in 1872, which can be considered the birth of the injection moulding process. Products of complex geometry necessitate the analysis of the mould filling process. There seems to be no published scientific investigation of this process until the experimental work of Gilmore and Spencer in 1950. This led to a review chapter by Beyer and Spencer in 1960 and mould filling experiments caught academic researchers' interest during the early 1960's. A decade later a new wave of mould filling flow visualization studies were reported and computer simulations began to emerge. This led to a better understanding of the mould filling process and in 1980 there was a breakthrough when Hieber and Shen [24] introduced the finite element method (FEM) to the field of injection moulding in their so-called

$2\frac{1}{2}$D approach. The key idea in the $2\frac{1}{2}$D approach is a combination of finite elements for solving the pressure field in two dimensions, where Hieber and Shen were utilizing a Galerkin FEM, and a finite difference method for the evaluation of temperature in the gapwise direction. Hieber and Shen also generalized the Hele–Shaw flow approach to the use of generalized Newtonian fluids; further the model was extended to the case of moulds of variable thickness. Since then the simulations have been extended to multi-component injection moulding, resin transfer moulding, gas assisted injection moulding, semi-crystalline materials, orientation of fibers in filled materials, viscoelastic models, and local phenomena at junctions and corners.

1.2. Hele–Shaw flow

The heat conductivity of thermoplastic polymers is low and since they solidify by cooling thin-walled products are dominating (wall thickness is typically 0.5–4 mm). This motivates the study of flow between parallel plates close together. The Hele–Shaw assumptions presuppose that the distance between the plates is much smaller than the dimensions of the flow. In the classical case one considers a Newtonian fluid, but the Hele–Shaw approximations have been used for the flow of non-Newtonian fluids, too. The latter is sometimes called generalized Hele–Shaw flow but we do not distinguish this from the classical case. The assumptions are:

- The flow is fully developed and laminar.
- The flow is incompressible.
- The velocity component in the gapwise direction is small compared to those parallel to the plates.
- The inertial forces resulting from the acceleration of the fluid are small compared to viscous shear forces.
- There is no slip at the plate surfaces.
- Velocity gradients parallel to the plates are small compared to those in the gapwise direction.
- Thermal conduction parallel to the plates is negligible compared to conduction in the gapwise direction.

The first assumption seems to be reasonable if we are not studying the flow too close to the fluid front or injection point. The others are standard assumptions, often used when analyzing Hele–Shaw flow of non-Newtonian fluids. For more details and further references one can consult [50]; pp. 89–92, 375–378 or [10], pp. 30–40. It follows from the assumptions and balance of momentum that the pressure is independent of z.

1.3. Weld lines and air traps

Weld lines occur when advancing flow-fronts "meet", i.e. when their outward normals are opposite or converging. For most polymers weld lines give rise to "witness marks", i.e. a visible defect in the surface finish. Since the flow at the front is of so-called fountain type and the particles at different meeting fronts have different mechanical history, they do not knit together completely and there is a mechanical weakness along the weld line. Weld lines occur behind obstacles (inserts) in the mould and when using more than one injection point. They can also be caused by variations of thickness of the mould. The position of the weld lines can be controlled or at least influenced by moving the injection point and/or varying the thickness of some region of the mould. Since the position of the weld lines could be very important it is important to predict these already when designing the product.

If the mould is to be filled properly the air in front of the flow-front must escape out of the mould. This is accomplished by introducing air vents at places in the mould that are last to be filled. If air gets surrounded by flow-fronts and has no way to escape we say that we have an air trap. Air traps are undesirable since the pressure that builds up increases the temperature locally so that the polymer will get burned and the product will be useless. Air traps can usually be avoided by suitable placement of air vents and injection points. It is also possible to modify the filling pattern by changing the thickness at some parts of the mould. It is obvious that there is a need for prediction of air traps and suggestions for placing air vents.

1.4. Simulation software

There are many commercial software products available for simulating injection moulding, e.g. Moldflow, I-DEAS, and C-MOLD. These are capable of computing with reasonable accuracy the filling pattern, weld lines, flow angle at "instant of fill", pressure distribution during packing stage, shrinkage, warpage, and temperature distribution and time to solidify when cooling. They are often part of a complete CAE-system, including a CAD-system for defining the geometry and structural analysis for computing deflection when the product is subjected to loads. This makes the software complex, both in its demands on computing power and the person operating the program. Despite the success in prediction, only a few percent of all companies in the plastics business use any of these software. The reason is probably the cost for the software/hardware and training of personnel to operate it.

For simulating the filling pattern most commercial software use the $2\frac{1}{2}$D approach (i.e. using finite elements in two dimensions) described in Section 1.1.3. Since the filling problem has a moving boundary this leads to problems with remeshing and tracking the boundary, or marking and filling cells. Detection of weld lines with good precision necessitates the use of small element sizes which increases the computation time. Small values for the power-law index give rise to a strong non-linearity that can lead to problems with convergence in the numerical algorithms. In situations where the Hele–Shaw assumptions are not satisfied (e.g. narrow channels), the accuracy of the simulations is very questionable.

The advantages of the commercial software packages are that they take many effects into account thus giving good accuracy in most cases and are a tool for analyzing the complete injection moulding process.

An overview of simulation software can be found in Manzione's book [34]. It is perhaps somewhat outdated, but contains interesting examples and can be useful as an introduction to the field of simulating injection moulding. A description of the flow analysis in Moldflow, taking an engineering approach, can be found in Kennedy's recent book [28].

There are also non-commercial simulators available, e.g. the simulator HeleShawFill in the Diffpack project at SINTEF, Oslo, Norway. The software is under development but could be a very interesting platform for experimentation with different numerical techniques. It is not yet designed for maximal performance but for flexibility, using an object oriented programming approach.

2. On the distance model as an asymptotic solution

A mathematical theory for Hele–Shaw flow of power-law fluids has been developed by Aronsson and Janfalk in [7]. This theory has then been applied to injection moulding of thermoplastic material of power-law type by Aronsson in [6] to give an asymptotic theory as the power-law index $n \to 0^+$. Hele–Shaw flow refers to the flow between two parallel plates, see Section 1.2. We will in this section give a brief description of the ideas and assumptions made. The basic result is the pseudo-circle principle, see Section 2.1.4. The idea of using the geometry information alone for determining the approximate filling pattern is not new, see e.g. [50], p. 384, Section 8.4.4 or [35]. The novelty by Aronsson is the mathematical derivation and the connection between the constitutive law and the distance concept, i.e. the pseudo-circle principle.

2.1. Some physical background

Introduce a Cartesian coordinate system such that the x and y axes are parallel to the plates and the flow cavity is contained between $z = -F(x, y)$ and $z = F(x, y)$, i.e. the gap between the plates is $2F(x, y)$. The function F is assumed to be slowly varying or a "step function" with discontinuities only at certain lines in Ω. The velocity components in the x, y, and z directions are denoted by u, v, and w. It is assumed that:

- The gap, $2F(x, y)$, is much smaller than the typical dimensions of the flow.
- The flow is fully developed and laminar.
- The fluid is generalized Newtonian and incompressible.
- The viscosity is given by a power law, see Section 1.1.2.
- The flow is isothermal or the viscosity is independent of temperature.
- There is no slip at the plate surfaces, i.e. $u = v = 0$ for $z = \pm F(x, y)$.
- The flow is such that viscous forces and pressure differences are dominating in the momentum equations. Thus, inertia and body forces are neglected.
- Due to geometric proportions and the no slip condition for $z = \pm F(x, y)$ the derivatives of u and v with respect to z are "normally" much bigger than derivatives with respect to x and y. Thus the contributions to the rate of strain tensor (and shear rate) from $\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, \frac{\partial v}{\partial x}$ and $\frac{\partial v}{\partial y}$ are neglected.
- The velocity is symmetric with respect to $z = 0$.
- All flow is parallel to the plates, i.e. $w = 0$. The z-component of viscosity forces is neglected. This is assumed to hold only at some distance away from the discontinuities of F.

The analysis is very much simplified by these assumptions. It can be found in [7], pp. 345–349; [50], pp. 375–378; [24], pp. 2–4, and in many other reports.

The approach for analyzing a continuum mechanical problem is to formulate the equations for balance of mass, momentum and energy and to combine them with a description of the material behaviour, a constitutive law. Granted the balance of linear momentum the symmetry of Cauchy's stress tensor is equivalent to balance of angular momentum. The constitutive law ensures the symmetry and we do not need the equations for balance of angular momentum explicitly.

Here we have assumed isothermal flow (or independence of temperature) and thus the energy equation is not needed. As mentioned in Section 1.2 the pressure is independent of z. By considering the averages

$$\overline{u}(x,y,t) = \frac{1}{2F(x,y)} \int_{-F(x,y)}^{F(x,y)} u(x,y,z,t)\,dz, \tag{2.3}$$

$$\overline{v}(x,y,t) = \frac{1}{2F(x,y)} \int_{-F(x,y)}^{F(x,y)} v(x,y,z,t)\,dz, \tag{2.4}$$

the problem may be reduced to two space dimensions. The flow is studied on a bounded Lipschitz domain $\Omega \subset \mathbb{R}^2$, which roughly means that corners are accepted but not cusps, spirals etc. Using the assumptions, the equation of motion (balance of linear momentum) and the continuity equation (balance of mass) integrated from $z = -F(x,y)$ to $z = F(x,y)$ become (after some manipulation)

$$\begin{pmatrix} \overline{u}(x,y,t) \\ \overline{v}(x,y,t) \end{pmatrix} = \overline{\mathbf{u}}(x,y,t) = -\nabla p(x,y,t)\frac{S(x,y,t)}{F(x,y)}, \tag{2.5}$$

$$\operatorname{div}\left(S(x,y,t)\nabla p(x,y,t)\right) = 0, \tag{2.6}$$

where

$$S(x,y,t) = \int_0^{F(x,y)} \frac{z^2}{\eta(\dot{\gamma}(x,y,z,t))}\,dz. \tag{2.7}$$

The equation (2.6) is known as Reynold's equation and S is the flow conductance (fluidity), see [50], p. 377. By inserting the power-law viscosity function and the velocity profile we can evaluate the integral and obtain S expressed in F, $|\nabla p|$, and material constants. Further, the velocity profile in the z-direction can be expressed in material constants, F, ∇p, and the z-coordinate. Averaging from $z = -F$ to $z = F$, the continuity equation (balance of mass) assumes the form

$$\operatorname{div}\left(F(x,y)^{2+\frac{1}{n}}|\nabla p(x,y,t)|^{\frac{1}{n}-1}\nabla p(x,y,t)\right) = 0. \tag{2.8}$$

We note that this is a q-harmonic equation with $q = 1 + \frac{1}{n}$, involving a weight. The above approach means that juncture losses (of pressure) are neglected.

Continuity of normal flow across a juncture is however implicit in the weak formulation of (2.8).

We denote the part of Ω which is filled by fluid at time t by Ω_t and let $\Gamma_0 = \partial\Omega_t \cap \Omega$, $\Gamma_1 = \partial\Omega_t \cap \partial\Omega$. The boundary conditions are $p = 0$ (after subtraction of a suitable constant) on Γ_0 and $\frac{\partial p}{\partial \mathbf{n}} = 0$ on Γ_1. Further, there is a kinematic boundary condition along Γ_0 expressing the conservation of mass. It says that in the outward normal direction the fluid front advances at a rate corresponding to the local rate of supply of fluid in that direction. The corresponding formula is not used in this derivation, the relation is implicitly used in deriving the pseudo-circle principle. Finally, the polymer is injected at some point $(x_1, y_1) \in \Omega$. The case of injection at the boundary $\partial\Omega$ or through several injection points is not considered here. For the injection point we have prescribed either an amount of polymer melt injected per second, or the pressure.

When considering injection at a point (x_1, y_1) we solve the equation (2.8) in $\Omega_t \setminus \{(x_1, y_1)\}$ with a prescribed singularity at (x_1, y_1). This can be seen as solving

$$\operatorname{div}\left(F(x,y)^{2+\frac{1}{n}}|\nabla p(x,y,t)|^{\frac{1}{n}-1}\nabla p(x,y,t)\right) = -c_1\delta_{(x_1,y_1)} \tag{2.9}$$

in the sense of distributions, where δ is the Dirac distribution. The physical interpretation is that the total flux across a circle $C \subset \Omega$ with center at (x_1, y_1) is prescribed.

We will now consider two problems:

1. The *instantaneous problem*, which is the problem of determining the pressure and flow fields at a certain moment, given the position of the flow front at that moment (and the rate of injection, or the pressure at the injection point).

2. The *evolution problem*, which is the problem of finding the filling pattern, i.e. the development with time of the fluid filled region Ω_t.

For the instantaneous problem we have equation (2.9) with the boundary conditions mentioned above, except the kinematic boundary condition. For the evolution problem, kinematic boundary data are also needed.

2.1.1. The instantaneous problem as an extremum problem

In the following we will write Ω for Ω_t, since we are only considering one fixed moment. Equation (2.9) is the Euler equation for the functional

$$J(p) = \frac{1}{1+\frac{1}{n}}\iint\limits_{\Omega} F(x,y)^{2+\frac{1}{n}}|\nabla p|^{1+\frac{1}{n}}\,dxdy - c_1 p(x_1,y_1).$$

The extremum problem P_q is to minimize $J(p)$ when $p \in U$, where U is a suitable class of functions, namely all $p \in W^{1,q}(\Omega)$ (where $q = 1 + \frac{1}{n}$) such that $p = 0$ on Γ_0.

2.1.2. The dual problem

Janfalk has in [27], pp. 347–349, applied the theory of Ekeland and Temam [18] and derived a dual convex problem. The exponent in our primary problem is $q = 1 + \frac{1}{n}$ and thus the conjugate exponent is $q' = 1 + n$ (where $\frac{1}{q} + \frac{1}{q'} = 1$ as usual). The dual problem $D_{q'}$ is to maximize the functional

$$J^*(\mathbf{r}) = -\frac{1}{1+n} \iint\limits_{\Omega} F(x,y)^{-(2n+1)} |\mathbf{r}(x,y)|^{1+n} \, dxdy$$

when the vector field $\mathbf{r}$ has components in $L^{q'}(\Omega)$ and satisfies

$$\iint\limits_{\Omega} \mathbf{r}(x,y) \cdot \nabla p(x,y) \, dxdy = -c_1 p(x_1, y_1) \tag{2.10}$$

for all $p \in U$. In the notation of [27], p. 348, we have $f = F^{2+\frac{1}{n}}$ and we obtain the exponent for F in the dual problem from the fact that $(2+\frac{1}{n})(1-q') = -(2n+1)$. The condition (2.10) means that $\operatorname{div} \mathbf{r} = c_1 \delta_{(x_1,y_1)}$ in the sense of distributions, where δ is the Dirac distribution.

Now it follows from the theory of convex analysis (see [27]) that the dual problem has a unique solution $\tilde{\mathbf{r}}$ satisfying

$$\tilde{\mathbf{r}} = -F^{2+\frac{1}{n}} |\nabla \tilde{p}|^{\frac{1}{n}-1} \nabla \tilde{p} \text{ a.e.}$$

where $\tilde{p}$ is the unique solution to the primary problem. Furthermore $J^*(\tilde{\mathbf{r}}) = J(\tilde{p})$.

2.1.3. The formal limit problem

Denote by $d((x,y),\Gamma_0)$ the *weighted interior distance* from (x,y) to Γ_0, i.e.

$$d((x,y),\Gamma_0) = \inf_{\gamma \in \Gamma((x,y),\Gamma_0)} \int_\gamma \frac{1}{F} \, ds$$

where $\Gamma((x,y),\Gamma_0)$ is the set of all rectifiable curves γ in Ω having one end-point on Γ_0 and the other end-point $= (x,y)$ and we integrate with respect to arc length. For two arbitrary points $(x_a,y_a), (x_b,y_b)$ in Ω we define the *weighted interior distance* analogously:

$$d((x_a,y_a),(x_b,y_b)) = \inf_{\gamma \in \Gamma((x_a,y_a),(x_b,y_b))} \int_\gamma \frac{1}{F} \, ds$$

As $n \to 0^+$ we suggest the following formal limit problems:

$$P_\infty : \text{ to find } \sup_p c_1 p(x_1, y_1)$$

when $p \in W^{1,\infty}(\Omega)$, $p = 0$ on Γ_0 and $||F\nabla p||_{L^\infty(\Omega)} \leq 1$; and

$$D_1: \text{ to find } \inf_{\mathbf{r}} \iint_\Omega \frac{1}{F(x,y)} |\mathbf{r}(x,y)| \, dxdy$$

when $\operatorname{div} \mathbf{r} = c_1 \delta_{(x_1,y_1)}$ in the sense of distributions, where δ is the Dirac distribution.

Clearly, P_∞ has the solution

$$p^\star(x,y) = d((x,y),\Gamma_0).$$

We introduce the set

$$E = \{(x,y) \in \Omega : d((x_1,y_1),\Gamma_0) = d((x_1,y_1),(x,y)) + d((x,y),\Gamma_0)\},$$

where (x_1,y_1) is the injection point.

Analogously to Theorem 2 by Aronsson [6], we have

Theorem 2.1. *For the problem P_∞, each solution $p(x,y)$ satisfies*

$$p(x,y) = d((x,y),\Gamma_0) \text{ for } x \in E.$$

It follows immediately that if $E = \Omega$, then $p^\star$ is the only solution to P_∞. It is also true that for any $(x,y) \in \Omega \setminus E$ there exists a solution p to P_∞ such that $p(x,y) \neq p^\star(x,y)$. Thus, E is the set of uniqueness for P_∞. See also [27], pp. 351–353.

Now we consider the family of dual problems $D_{q'}$ from Section 2.1.2 with $1 < q' < 2 \quad (q' = 1+n)$ and the formal limit problem D_1. We denote the unique solution of $D_{q'}$ by $\mathbf{r}_{q'}$. If we forget about the choice of spaces we note that $D_{q'}$ means minimizing

$$J_{q'} = \iint_\Omega F^{1-2q'} |\mathbf{r}|^{q'} \, dxdy$$

and D_1 means minimizing

$$J_1 = \iint_\Omega \frac{1}{F} |\mathbf{r}| \, dxdy$$

under the same constraint, namely that $\operatorname{div} \mathbf{r} = c_1 \delta_{(x_1,y_1)}$. We interpret the integrals $J_{q'}$ and J_1 as the "cost" for the transportation of a fluid from (x_1,y_1) to Γ_0. When $q' > 1$ there is an extra cost for concentrating the flow and at the optimal $\mathbf{r}_{q'}$ there is a balance between this cost and the gain by letting the fluid flow along shorter paths (according do the weighted metric) to Γ_0. In the limit case ($q' = 1$) there is no extra cost for concentrating the flow, so it is intuitively clear that the fluid will flow along shortest paths from (x_1,y_1) to Γ_0. Janfalk has proved a thinning theorem in [27], pp. 357–359, stating that if $E \neq \overline{\Omega}$, then the optimal $\mathbf{r}'_q \to 0$ in the mean as $q' \to 1+$, on sets "away" from E. Note however that the weight function f in [27] does not depend on q' and occurs as $f^{1-q'}$ in the dual problem while our weight $F^{2+\frac{1}{n}}$ which in the dual problem occurs as $F^{-(2n+1)}$ does depend on q' since $n = q' - 1$. The consequence is that the weight function "survives" in the limit process and the distance concept has to be modified as indicated above. One can modify Janfalk's thinning theorem and proof so that it covers this case, too. A meaningful limit process here for injection moulding should therefore be such that Ω_t expands only near points on $\partial\Omega_t$ closest to (x_1, y_1) in the d metric. But then Ω_t must be a family of pseudo-circles as shown by Aronsson in [6]. Thus we have the following pseudo-circle principle:

2.1.4. The pseudo-circle principle

For small values of the power-law index n the fluid filled region expands approximately like a family of pseudo-circles in Ω with respect to the metric $F^{-1}\, ds$, each pseudo-circle having its center at the injection point (x_1, y_1).

3. On the distance function in the case of constant thickness

3.1. Preliminaries

Let $\Omega \subset \mathbb{R}^2$ be a bounded polygonal Lipschitz domain, let $s \in \overline{\Omega}$ be a fixed point called the starting point, and let N be the number of reentrant[1] corners on $\partial\Omega$. The restriction to reentrant corners is in not necessary to obtain a working algorithm but can, in some cases, reduce the computation time. Denote the set of rectifiable curves from s to $g \in \overline{\Omega}$ contained in $\overline{\Omega}$ by M_g, and denote the Euclidean length of a curve $\gamma \in M_g$ by $L(\gamma)$. Define the distance function d as

$$d(g) = \inf_{\gamma \in M_g} L(\gamma).$$

A curve $\gamma \in M_g$ such that $L(\gamma) = d(g)$ is called a Euclidean shortest path. The connected components of $\complement\Omega$, including the unbounded component if it has reentrant corners in Ω, are called obstacles. It is well known that a shortest path must be a polygonal curve with its corners on the obstacle vertices coinciding with reentrant corners of $\partial\Omega$. The existence of a shortest path is also well known, see e.g. [11], p. 25, 5.18.

3.2. Results of others

The problem of computing a Euclidean shortest path in a planar polygonal domain Ω is a well known problem in computational geometry; see e.g. Lozano-Pérez and Wesley in [32]. It is usually called the Euclidean shortest path problem. The problem has obvious applications in routing and robotics and has drawn the attention of many researchers. Often these problems have straightforward generalizations to the weighted case (see Section 4 for a definition of weighted case) by considering some non-constant cost per unit travel in the domain.

It is possible to decompose the problem into two parts; the first is to construct the visibility graph for reentrant corners of $\partial\Omega$ (including the injection point as a node in the graph) and the second to find a shortest path in that graph. This decomposition is not necessary and there are some recent algorithms that do not use this decomposition, see Section 3.2.3. As we will see below the best algorithm using the decomposition found so far runs in $O(E + N \log N)$ time, where E is the number of edges in the graph which is $O(N^2)$. When we know a suitable shortest path tree of the visibility graph we can find a shortest path from s to an arbitrary point $t \in \overline{\Omega}$ easily, namely by minimizing the sum of the lengths of the line segment to a visible node in the graph (or s if s is visible from t) and a shortest path from that node to s. The minimization runs over all nodes visible from t. Without using the decomposition the best algorithms known so far are solving the problem in $O(N \log^2 N)$ time and $O(kN + N \log N)$ time, where k is the number of obstacles.

[1] The angle within Ω is $> \pi$.

3.2.1. Constructing the visibility graph

The following definition, slightly modified by the author of this thesis, was given by Mitchell in [36].

Definition 1. The *visibility graph* is a graph whose nodes are identified with the reentrant corners of $\partial\Omega$ and the injection point. Further, the graph has edges joining pairs of nodes that "see" each other (i.e. the line segment joining the nodes does not intersect the interior of any obstacle).

The paper [36] by Mitchell is a good survey of computational geometry methods for route planning problems. It contains a survey of algorithms for constructing the visibility graph which we will summarize below. It further mentions other methods for computing shortest paths that we will also briefly touch upon. Further, one can find several generalizations of the Euclidean shortest path problem in Mitchell's paper, including the WRP.

We note that a visibility graph with N nodes has at most $\binom{N}{2} = \frac{N(N-1)}{2}$ edges.

A straightforward algorithm for computing the visibility graph is to check each pair of nodes (u, v) for visibility by testing if the interior of the line segment $\overrightarrow{uv}$ crosses the boundary of the domain. This process has time complexity $O(N^3)$.

A better algorithm can be obtained by constructing the visibility polygon around each node v by sweeping a ray $r(\theta)$ around v. One uses an angularly sorted list of the nodes to update a list of the obstacle segments intersecting the ray. This gives an algorithm that finds the visibility graph in $O(N^2 \log N)$ time, using $O(N)$ working space.

In 1985 Welzl in [51] (and, independently in 1986, Asano et al in [8]) gave algorithms which run in $O(N^2)$ time. The new idea is to map the nodes on their dual lines and build the arrangement (see Edelsbrunner et al in [17]) of these lines (in $O(N^2)$ time) to obtain the sorted order of nodes about each node v in total time $O(N^2)$. Further, the arrangement gives the possibility of performing a topological sorting to guide the construction of the visibility polygon for every node.

In 1989 Edelsbrunner and Guibas in [15], with corrigendum in [16] (a new proof of Theorem 3.1), were able to find the visibility graph in $O(N^2)$ time requiring $O(N)$ working space using the notion of topological sweep.

Since the visibility graph in the worst case could contain $\Omega(N^2)$ edges this also gives the worst case behaviour of an algorithm for constructing it. But since the visibility graph could be sparse in some cases one could argue for measuring the algorithm complexity in an output sensitive way, i.e. measuring the time proportional to the size of the output. Here the output is the number E of edges in the visibility graph. Ghosh and Mount have in [21] presented such an algorithm with running time $O(E + N \log N)$ using $O(E)$ working space.

3.2.2. Finding a shortest path in a graph

This is a classical problem in network optimization that has been extensively studied. In 1959 Dijkstra described in [13] an important algorithm that works for graphs with nonnegative costs. This was one of the first computationally efficient algorithms for this problem. For a modern exposition of the subject one can consult the book by Evans and Minieka [19], pp. 77–122. Some historical remarks as well as a somewhat sceptical survey of the literature up to 1968 can be found in Dreyfus' paper [14]. Dijkstra's algorithm runs in time $O(N^2)$. In some cases an alternative to Dijkstra's algorithm is the so-called A^* algorithm, as pointed out by Mitchell in [36]. The A^* algorithm also runs in $O(N^2)$ time but is believed to have better average case behaviour when one wants to find the shortest path between a pair of nodes. Since we want to find the distance from one node to all the other nodes in the graph the A^* algorithm will not improve the performance in our case. By using a Fibonacci heap data structure for storing the non-labeled nodes in Dijkstra's algorithm Fredman and Tarjan in [20] obtained an algorithm running in $O(E + N \log N)$ time. By using a priority queue (see Knuth [30]) one can get an algorithm that runs in $O((E + N) \log N)$ time, but is easier to implement and in practice has been found to be competitive by e.g. Hung and Divoky in [25] and Cherkassky, Goldberg, and Radzik in [12]. For a description of this variant of Dijkstra's algorithm one can consult Gibbons' book [22], pp. 13, 36–38.

3.2.3. Shortest path map algorithms

We will now introduce some terminology needed for describing a shortest path map and algorithms based on that approach. The first definition was given by Mitchell in [36] and [37].

Definition 2. A point $r \in \overline{\Omega}$ is called a *root* of a point $t \in \overline{\Omega}$ if, for some shortest path from s to t, r is the last corner along this path. If the line segment $\overrightarrow{st}$ is a shortest path then s is a root of t.

This is almost the same as the concept *foot-point* of t defined by Aronsson in [5]. That definition is as follows:

Definition 3. Let $x \in \overline{\Omega}$. Let $V(x)$ denote the set of concave corners y of Ω such that y is visible from x within $\overline{\Omega}$; plus the point s if s is visible from x. A *foot-point* of x is a point $y \in V(x)$ such that $d(x) = d(y) + L(\overrightarrow{xy})$.

The relations between the two definitions are as follows:

Observation 3.1. *Let r be a root of t. A concave corner of Ω on the line segment $\overrightarrow{rt}$ is a foot-point of t. Also r is a foot-point of t.*

Observation 3.2. *Let y be a foot-point of t. Then y is a root of t if some shortest path to t has a corner at y or if $y = s$. Note that a foot-point of t is not necessarily a root of t.*

Mitchell has in [36] partitioned $\overline{\Omega}$ into maximal regions consisting of points having the same root or set of roots and he calls this partition a shortest path map. This map can be built from a shortest path tree in $O(N \log N)$ time. Early algorithms for constructing the shortest path map directly (i.e. without constructing the visibility graph, which in worst case consumes $\Omega(N^2)$ working space) have $\Omega(N^2)$ running times in the worst case. Recently Mitchell published in [38] an algorithm running in $O(N^{5/3+\epsilon})$ time and space by using some advanced range searching data structures to locate the vertices of the shortest path map. Hershberger and Suri have obtained in [23] an algorithm running in $O(N \log^2 N)$ time using $O(N \log N)$ working space. They used a technique called the continuous Dijkstra method, introduced by Mitchell, Mount and Papadimitriou in [40] for solving the discrete geodesic problem, which is to find a Euclidean shortest path on a polyhedral surface. Hershberger and Suri also mention that the Euclidean shortest path problem has a lower bound of $\Omega(N \log N)$ on the running time in the algebraic computational tree model. Storer and Reif have in [47] obtained an algorithm which runs in $O(kN+N \log N)$ time and uses $O(N)$ working space, in the case of k obstacles.

3.2.4. Properties of the distance function and interpretation of the results in injection moulding applications

Aronsson has in [5] proved the following lemma:

Lemma 3.3. *Let $y \in \Omega, y \neq s$ and put $t = d(y)$, $C_r = \{x : |x - y| < r\}$, and $\Omega_t = \{x : x \in \Omega, d(x) \leq t\}$. Then $\underline{\lim}_{r \to 0^+} \frac{m(\Omega_t \cap C_r)}{m(C_r)} \geq \frac{1}{2}$, where m denotes the Lebesgue measure.*

From this Aronsson obtains the corollary that Ω_t can not have a convex corner in Ω.

The concept of weld lines was described in Section 1.3 and will be defined mathematically for the distance model in Definition 12 in Section 4.6.2. From Aronsson's analysis in [5] it follows that in the case of constant thickness weld lines correspond to the locus of discontinuities of ∇d. When a point $x \in \Omega$ has at least two foot-points y_1 and y_2 such that $\overrightarrow{xy_1}$ and $\overrightarrow{xy_2}$ are not parallel it is called a *switch point* and d is not differentiable at x. Conversely, at an arbitrary point $x \in \Omega$ which is not a switch point d is differentiable. Further, the set S of shift points is relatively closed in Ω and $d \in C^1(\Omega \setminus S)$. Moreover, Ω is simply connected if and only if $S = \emptyset$. Note that $S = \emptyset$ is equivalent to $d \in C^1(\Omega)$. For details see [5]. The set S constitutes the weld lines and consists of a finite number of closed subarcs of hyperbolas (including the case of a straight line) with foci at concave corners of Ω (i.e. the angle in Ω is $> \pi$).

Let $t > 0$ be given. The level set

$$\{x : x \in \Omega, d(x) = t\}$$

corresponds to the position of the flow-front at some instant of time, see Section 8.1 and [6]. In [5] Aronsson showed that this level set consists of (at most) a finite number of circular arcs, plus (at most) a finite number of isolated points. Each of the arcs has its center at a concave corner of Ω or the injection point.

3.2.5. Implementations for injection moulding applications

In the injection moulding application constant weight corresponds to constant thickness of the mould. Algorithms for finding the shortest path by constructing the visibility graph and finding the shortest path in a graph with Dijkstra's algorithm have been implemented for injection moulding applications by Aigner in [2], for planar polygonal domains, and by Lundquist in [33], a modified variant for polyhedral surfaces.

The straightforward algorithm for constructing the visibility graph running in $O(N^3)$ time is the one implemented by Aigner. Lundquist also has essentially this approach, but he observes that it is enough to consider visibility among reentrant corners.

4. The weighted region problem, WRP

4.1. Preliminaries

The name WRP for the problem of finding a path with shortest weighted length in a planar subdivision is adopted from Mitchell and Papadimitriou [41] and we define it in the following context:

Let $\Omega \subset \mathbb{R}^2$ be a bounded polygonal Lipschitz domain with a subdivision of its closure $\overline{\Omega} = \cup P_i$ such that for $i = 1, \ldots, n$, P_i is a closed and convex polygon with nonempty interior $\mathring{P}_i$ satisfying

$$\mathring{P}_i \cap \mathring{P}_j = \emptyset, \quad \text{if } j \neq i;$$

furthermore for all $i > n$ and all $i < 1$ we define $P_i = \emptyset$ (this is only for convenience in notation). A point $x \in \overline{\Omega}$, can be contained in more than one polygon but we make the following definition to select one of them uniquely:

Definition 4. The *polygonal reference function*, $p : \overline{\Omega} \to \{1, \ldots, n\}$, is defined by

$$p(x) = \min\{i : x \in P_i\}.$$

Let $\{w_i\}_{i=1}^n \subset \mathbb{R}$ be such that

$$0 < w_1 \leq w_2 \leq \ldots \leq w_n.$$

Definition 5. The *weight function*, $w : \overline{\Omega} \to \mathbb{R}$, is defined by $w(x) = w_{p(x)}$.

Note that

$$w(x) = \min\{w_i : x \in P_i\},$$

i.e. the unique polygon that is referred to by p has the least weight of all polygons containing x. This definition also means that w is piecewise constant and lower semi-continuous. Let $s, g \in \overline{\Omega}$. Denote by M the set of rectifiable curves from s to g and contained in $\overline{\Omega}$.

Definition 6. The *weighted length*, $L_w : M \to \mathbb{R}$, is defined for a curve $\gamma \in M$ by

$$L_w(\gamma) = \int_\gamma w \, ds,$$

where we integrate with respect to arc length.

We denote the Euclidean length of a curve $\gamma \in M$ by $L(\gamma) = \int_\gamma ds$. Note that parameterization with respect to arc length is absolutely continuous.

4.2. Statement of the problem

The weighted region problem is to show that there exists $\gamma^\star \in M$ such that

$$L_w(\gamma^\star) = \inf_{\gamma \in M} L_w(\gamma),$$

to describe the properties of $\gamma^\star$ and to construct an algorithm for computing an approximation of $L_w(\gamma^\star)$. Sometimes one also wants an approximation of $\gamma^\star$.

4.3. Results of others

First we note that the restriction that the polygons P_i in the subdivision have to be convex is not necessary, but simplifies the algorithm. To be more specific, in the step 5 in Section 5.2.2 we need for each polygon only store the set A_l of discretization points on that polygon, since the polygon is convex and all points on its boundary can "see" each other. In the non-convex case visibility checking needs to be added in step 2 and 3a in Section 5.2.3. By using convex polygons, we also get a simple situation in the proof of Theorem 5.1. There we want to perturb the line segment passing through a polygon and when the polygon is convex the perturbed line segment will still remain in the polygon when the perturbation is small enough. This may not be the case when the polygons are not convex.

Early algorithms for the WRP used a uniform tessellation of Ω and associated for each cell in the tessellation a weight corresponding to the weight of the polygon containing (the major part of) the cell. Then the algorithm expanded a wavefront about the starting point s at rates proportional to the reciprocal of the weights. The algorithm stopped when the goal point g was reached, and back pointers were stored to recover the path supposed to approximate a shortest one. A discussion of the difficulties connected to such an algorithm can be found in [45], [39] and [42]. One problem with that approach is the digitization bias created by confining movements to a fixed number (usually 4 or 8) of orientations. In the non-weighted case an almost error free[2] wavefront propagation algorithm has been developed by e.g. Ragnemalm in his PhD thesis, [44] pp. 121–167. The idea is to store, for each cell in the tessellation, the last line segment on a shortest path to the cell. The cells are traversed, and the line segments constructed, by propagating a wavefront through the domain. When propagating the wavefront one must keep track of the propagation direction for different segments of the wavefront. It is not obvious how to extend this procedure to the weighted case.

However, Mitchell and Papadimitriou have in [41] presented an algorithm for the WRP, based on the continuous Dijkstra method introduced by Mitchell, Mount and Papadimitriou in [40] for solving the discrete geodesic problem. It runs in $O(N^8 L)$ time using $O(N^4)$ working storage, where N is the number of vertices in the subdivision of $\overline{\Omega}$ and L is the precision of the problem, i.e. the number of bits needed to encode the problem. In particular we have $L = O(\log(NKv_k/v_1\epsilon))$, where K is the maximal integer coordinate (when encoding the problem in positive integer coordinates) of any vertex in the subdivision; the weights are assumed to be integers and $\epsilon > 0$ is the error tolerance. The algorithm will find a path $\hat{\gamma}$ such that $L_w(\hat{\gamma}) \leq (1+\epsilon)L_w(\gamma^*)$. However, that algorithm was designed for complexity analysis rather than real computations. We do not know of any implementation of this algorithm.

At the same time Rowe and Richbourg, independently of Mitchell and Papadimitriou, see [45], developed and implemented an algorithm for solving the WRP. Their algorithm has not been proven better than exponential in the worst

[2] See discussion in [44], pp. 147–148.

case, but for a terrain with a "statistically even" distribution of features they have an average-case running time and working storage of $O(N^2)$.

This algorithm is designed for practicality and they have implemented a variant using seven pruning methods considered most cost effective out of the eleven that were given in [45].

To briefly describe the main idea in the algorithm we need a definition.

Definition 7. A well-behaved path subspace (WBPS) is a maximal collection of polygonal curves emanating from a point x such that they

1. cross the same ordered sequence of separation lines (i.e. interior edges of the polygons in the subdivision of $\overline{\Omega}$),
2. obey the local optimality criterion (Snell's Law) at the separation lines,
3. have their corners on the separation lines, and
4. do not pass through any vertex of a polygon in the subdivision.

A wedge is a WBPS plus the paths at the boundaries to the neighbouring WBPSs.

Let S be a set of all polygonal curves emanating from s, crossing[3] a given sequence of separation lines, and then heading straight to g after crossing the last separation line (an edge on the boundary of the polygon containing g). I.e. S is essentially a WBPS without the local optimality condition 2 above and with the additional requirement that the curves end up at the point g. We can specify a path in S by giving, for each separation line, the distance from one fixed end point of the separation line to the intersection between the path and the separation line, as a parameter. The weighted length is a convex function of those parameters, as is shown in [45], but since it is not strictly convex this does not imply that any local minimum is unique, contrary to the statement in [45]. For a counterexample see Figure 2. However, one can make a systematic

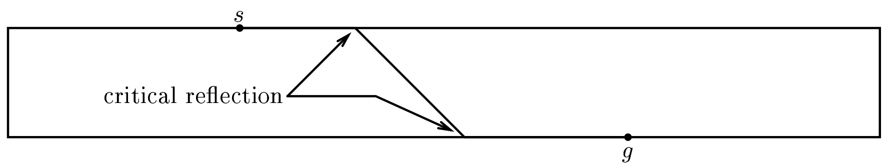


Figure 2: Example of a WBPS with a whole family of minima.

search of the WBPSs to find an optimal path. The algorithm consists of finding a wedge that contains an optimal path from s to g; then the actual path within the wedge can be found by some iterative numerical search method. The WBPSs can be found by observing that the bounding curves are always locally optimal paths that pass through at least one vertex in the subdivision of Ω. Hence the WBPSs can be found by finding the paths obeying Snell's Law and going from s to a vertex in the subdivision. For full details we direct the reader to [45].

[3] At a point in the relative interior of the separation line.

4.4. Existence and properties of $\gamma^\star$

First we need to characterize some special points on polygonal curves in our weighted region.

Definition 8. A point $x \in \gamma$, where $\gamma \in M$ is a simple polygonal curve, is a *shift point* for γ if for every neighbourhood N of x there exist two points $y_1, y_2 \in N \cap \gamma \setminus \{x\}$ such that $p(y_1) \neq p(y_2)$.

Observation 4.1. *Consider the function $f(t) = p(x(t))$ for a simple polygonal curve $\gamma \in M$ with parameterization $x(t)$ with respect to arc length. (There is no need to distinguish between the parameter value t and the point $x(t)$.) Then*

1. *The function f is piecewise constant.*
2. *Each point t^*, for which there does not exist a neighbourhood N of the point such that $f|_N$ is constant is either*
   (a) *a shift point, or*
   (b) *such that there exists a neighbourhood, N_0, of the point such that $f|_{N_0 \setminus \{t^*\}}$ is constant, c. This case occurs only if the point is*
      i. *an endpoint of the curve and situated on the boundary of a polygon with index $i < c$, or*
      ii. *a corner of γ on the boundary of such a polygon, or*
      iii. *a vertex of such a polygon, or*
      iv. *both* 2(b)ii *and* 2(b)iii*.*

   Clearly the endpoints of γ are not shift points.

When studying shortest paths we need not bother about the case 2(b)ii (and consequently neither the case 2(b)iv), as will be shown later in Lemma 4.4. Denote by $\{v_i\}_{i=1}^k$ the different weights in $\{w_i\}_{i=1}^n$ and n_i the number of weights with value v_i, ordered so that

$$\begin{aligned}
0 &< v_1 < v_2 < \dots < v_k, \\
v_1 &= w_1 = w_2 = \dots = w_{n_1}, \\
v_2 &= w_{n_1+1} = w_{n_1+2} = \dots = w_{n_1+n_2}, \\
&\vdots \\
v_k &= w_{n_1+n_2+\dots+n_{k-1}+1} = w_{n_1+n_2+\dots+n_{k-1}+2} = \dots = w_n, \\
n &= \sum_{0<i\leq k} n_i.
\end{aligned}$$

In Mitchell, Mount and Papadimitriou [40], Lemma 3.2, p. 650 it is observed that a shortest path in the Discrete Geodesic Problem (DGP) intersects a convex face in a line segment or not at all. An analogous reasoning gives in our situation:

Observation 4.2. (Simple stretching operation) *If $\gamma \in M$ and*

$$w_j = \min\{w(x) : x \in \gamma\}$$

then there is a $\gamma' \in M$ such that $L_w(\gamma') \leq L_w(\gamma)$ and $\gamma' \cap P_j$ is empty, one point or a line segment. We say that we obtain γ' by stretching γ into P_j.

Proof: If $\gamma \cap P_j = \emptyset$ we are done. Otherwise, let x_1 be the first point on $\gamma \cap P_j$ and x_2 the last point on $\gamma \cap P_j$. If $x_1 = x_2$ we are done. Otherwise, denote the part of γ between x_1 and x_2 by γ_{12}. Note that

$$\int_{\gamma_{12}} w\, ds \geq \int_{\gamma_{12}} w_j\, ds \geq L(\overrightarrow{x_1 x_2}) w_j.$$

By replacing γ_{12} by the line segment $\overrightarrow{x_1 x_2}$ we obtain γ', which completes the proof. $\square$

We will use the following algebraic identity, where $i, j, k \in \mathbb{Z}$, when determining an upper bound for the number of shift points on $\gamma^\star$. We will follow the usual convention and interpret an empty sum as 0 and an empty product as 1.

Lemma 4.3.

$$\sum_{0<i\leq k} \left(n_i \prod_{0<j<i} (n_j + 1) \right) = \prod_{0<i\leq k} (n_i + 1) - 1, \text{ for all } k \tag{4.11}$$

Proof: Induction over k. If $k < 1$, it is trivial that (4.11) holds. If

$$\sum_{0<i<k} \left(n_i \prod_{0<j<i} (n_j + 1) \right) = \prod_{0<i<k} (n_i + 1) - 1, \text{ for some } k > 0, \tag{4.12}$$

which is the induction hypothesis, then

$$\begin{aligned} &\sum_{0<i<k} \left(n_i \prod_{0<j<i} (n_j + 1) \right) + n_k \prod_{0<j<k} (n_j + 1) \\ &= \left(\prod_{0<i<k} (n_i + 1) \right) (1 + n_k) - 1 = \prod_{0<i\leq k} (n_i + 1) - 1. \end{aligned}$$

The principle of induction gives that (4.11) is true. $\square$

We are now ready for the following useful lemma:

Lemma 4.4. (Fundamental Reduction Lemma) *For every $\gamma \in M$ there exists a polygonal curve $\gamma' \in M$ with at most*

$$C = \prod_{0<i\leq k} (n_i + 1) - 2$$

shift points, and such that $L_w(\gamma') \leq L_w(\gamma)$ with strict inequality, unless $\gamma' = \gamma$. Finally, γ' is obtained from γ by successive stretching as in Observation 4.2.

Proof: We will use the simple stretching operation recursively; first for the "cheapest" polygons, then for the next to cheapest, etc. Let $\gamma'_0 = \gamma$. For $j = 1, \ldots, n_1$ we recursively denote the curve obtained by stretching γ'_{j-1} into P_j by γ'_j. Then

$$\gamma'_{n_1} \cap \left(\bigcup_{i \leq n_1} P_i \right)$$

consists of at most n_1 line segments (and maybe some isolated points). By removing these line segments from γ'_{n_1} we get a family of curves, G_1 say, with $\#G_1 \leq n_1 + 1$. We add the endpoints to these curves. Now we repeat this procedure at the second level (stretching into next to cheapest polygons) for each curve in G_1. Observing that the minimum weight on each curve in G_1 is at least v_2 (maybe except for some isolated points) we can still use the stretching trick. Note that the stretching has to be done in order so that we for each curve in G_1 first let stretching into P_{n_1+1} define a new curve which is then stretched into P_{n_1+2}, etc. Then we have at most $n_1 + 1$ curves, each having at most n_2 line segments in polygons with weight v_2. By removing these at most n_2 line segments from each of the at most $n_1 + 1$ curves we get a family of curves, G_2 say, with $\#G_2 \leq (n_2 + 1)(n_1 + 1)$. Possibly apart from endpoints (added) we get curves contained in

$$\bigcup_{i > n_1 + n_2} P_i.$$

By reasoning in this way for each of the k different weights in increasing order we get at most

$$n_k \prod_{0<j<k} (n_j + 1)$$

line segments at the highest level. When we remove these line segments there is nothing left and we have replaced γ by γ', consisting of at most

$$\sum_{0<i\leq k} \left(n_i \prod_{0<j<i} (n_j + 1) \right)$$

line segments, such that $L_w(\gamma') \leq L_w(\gamma)$. There are no shift points for γ' on the interior of the line segments since each is contained in one polygon. Further, there can not be two consecutive line segments contained in one polygon due to the reduction, thus the case 2(b)ii in Observation 4.1 can not occur and the polygonal curve is determined by its shift points. We can draw the conclusion that γ' has at most

$$\sum_{0<i\leq k} \left(n_i \prod_{0<j<i} (n_j + 1) \right) - 1 = \prod_{0<i\leq k} (n_i + 1) - 2$$

shift points, where the equality follows from Lemma 4.3. Finally, $L_w(\gamma') = L_w(\gamma)$ if and only if we do not modify γ during the reduction procedure. $\square$

We can immediately observe some facts concerning positions of shift points.

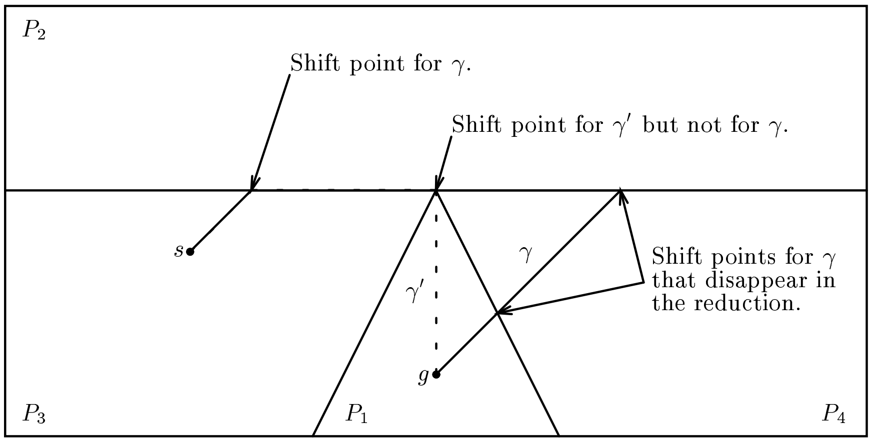


Figure 3: Example where two shift points for γ are "moved and joined" to become one shift point for γ'.

Observation 4.5. *If γ is a polygonal curve (so the concept of shift points is defined for γ) then γ' does not have more shift points than γ.*

Observation 4.6. *If γ is a polygonal curve, then a shift point for γ' is not necessarily a shift point for γ; see Figure* 3.

Proposition 4.7. *A shortest path exists, but is is not necessarily unique. Each shortest path is invariant under reduction as in Lemma* 4.4*, and is thus a polygonal curve and has at most C shift points.*

Proof: Let

$$d = \inf_{\gamma \in M} L_w(\gamma).$$

We call d the *weighted interior distance from s to g.* There exists a sequence

$$\{\gamma_p\}_{p=1}^{\infty} \subset M$$

such that $L_w(\gamma_p) \to d,\ p \to \infty$. For $p = 1, 2, \ldots$ we apply Lemma 4.4 to γ_p and get polygonal curves γ_p' having at most C shift points. By successive selections we can get subsequences such that the number of shift points is constant and such that the first, second, etc. shift point converges. Let $\gamma^\star$ be the polygonal curve through the limits of shift points. Since w is lower semi-continuous it follows that $L_w(\gamma^\star) = d$. If we apply Lemma 4.4 to a shortest path we can not get $L_w(\gamma') < L_w(\gamma)$, so $\gamma = \gamma'$ for shortest paths and the theorem follows. $\square$

4.5. Uniqueness

First we will study the length u of a line segment U as a function of the endpoints of U, moving along the interior of two non-collinear edges with disjoint interior. Further, the two edges should be situated on the boundary of the same convex polygon P_i that in the sequel will be one of the polygons in the subdivision mentioned above. The coordinate for an endpoint of U is given as the Euclidean distance from one of the endpoints of the edge in question, see Figure 4. The edges may be parallel.

We have that $s_1 > 0$ and $s_2 > 0$ and thus $u > 0$.

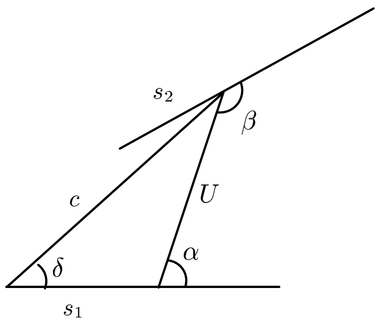


Figure 4:

We have that $u^2 = s_1^2 + c^2 - 2s_1 c\cos\delta$ and $c\cos\delta - s_1 = u\cos\alpha$. Holding s_2 fixed we have δ and c constant and we easily obtain

$$\frac{\partial u}{\partial s_1} = -\cos\alpha,$$

and similarly

$$\frac{\partial u}{\partial s_2} = -\cos\beta.$$

If we again hold s_2 fixed, then $u\sin\alpha$ is constant and thus

$$\frac{\partial u}{\partial s_1}\sin\alpha + u\cos\alpha\frac{\partial\alpha}{\partial s_1} = 0.$$

If $\alpha \neq \frac{\pi}{2}$ then

$$\frac{\partial\alpha}{\partial s_1} = -\frac{\frac{\partial u}{\partial s_1}\sin\alpha}{u\cos\alpha} = \frac{cos\alpha\sin\alpha}{u\cos\alpha} = \frac{\sin\alpha}{u}.$$

It is easy to verify that

$$\frac{\partial\alpha}{\partial s_1} = \frac{1}{u}$$

when $\alpha = \frac{\pi}{2}$ and thus

$$\frac{\partial\alpha}{\partial s_1} = \frac{\sin\alpha}{u}$$

for all s_1 and s_2 in question. Similarly

$$\frac{\partial \beta}{\partial s_2} = \frac{\sin \beta}{u}.$$

Since $\alpha + \beta$ is constant, we have

$$\frac{\partial \beta}{\partial s_1} = -\frac{\partial \alpha}{\partial s_1} = -\frac{\sin \alpha}{u}.$$

Now we obtain

$$\begin{aligned}
\frac{\partial^2 u}{\partial s_1^2} &= \sin \alpha \frac{\partial \alpha}{\partial s_1} = \frac{\sin^2 \alpha}{u} \\
\frac{\partial^2 u}{\partial s_1 \partial s_2} &= \sin \beta \frac{\partial \beta}{\partial s_1} = -\frac{\sin \alpha \sin \beta}{u} \\
\frac{\partial^2 u}{\partial s_2^2} &= \sin \alpha \frac{\partial \beta}{\partial s_2} = \frac{\sin^2 \beta}{u}
\end{aligned}$$

Taking an arbitrary line in parameter space

$$\begin{aligned}
s_1 &= \sigma_1 + \lambda t \\
s_2 &= \sigma_2 + \mu t
\end{aligned}$$

we have

$$\frac{du}{dt} = \frac{\partial u}{\partial s_1}\lambda + \frac{\partial u}{\partial s_2}\mu$$

and

$$\begin{aligned}
\frac{d^2 u}{dt^2} &= \frac{\partial^2 u}{\partial s_1^2}\lambda^2 + 2\frac{\partial^2 u}{\partial s_1 \partial s_2}\lambda\mu + \frac{\partial^2 u}{\partial s_2^2}\mu^2 \\
&= \frac{1}{u}\left(\lambda^2 \sin^2 \alpha - 2\lambda\mu \sin \alpha \sin \beta + \mu^2 \sin^2 \beta\right) \\
&= \frac{1}{u}\left(\lambda \sin \alpha - \mu \sin \beta\right)^2 \geq 0.
\end{aligned}$$

Thus u is a convex function. Further

$$\frac{d^2 u}{dt^2} > 0$$

unless

$$\lambda \sin \alpha - \mu \sin \beta = 0. \tag{4.13}$$

When (4.13) holds we have

$$\frac{d\alpha}{dt} = \frac{\partial \alpha}{\partial s_1}\lambda + \frac{\partial \alpha}{\partial s_2}\mu = \frac{1}{u}(\lambda \sin \alpha - \mu \sin \beta) = 0$$

and thus also

$$\frac{d\beta}{dt} = 0.$$

Thus α and β are constant in this type of motion, i.e. the segment U moves parallel to itself and the length u varies linearly with t. This is the only case where $u = u(t)$ is not strictly convex. Note that when the edges are parallel $\alpha = \pi - \beta$ and thus $\lambda = \mu$ in the case of linearity.

Using the notion of a *window sequence* (WS) defined by Kindl et. al. [29] we will investigate uniqueness of a shortest path in a suitable context, i.e. within a WS. First, we note that the claim 5. in [29], pp. 9, that a *unique* locally optimal path must exist within each WS is not true; consider e.g. Figure 2. The concept of a WS is in some sense present in [45], there named wedge, and in [41] as an edge sequence and the concept of a path connecting an edge sequence, but here we will only use the concept of WS.

Definition 9. Given the edges $e_1, \dots, e_k$ where

1. two consecutive edges are part of the boundary of a common convex polygon of the subdivision of $\overline{\Omega}$,
2. two consecutive edges have disjoint interior,
3. two consecutive edges are not collinear,
4. s is not collinear with e_1, and
5. g is not collinear with e_k

we define a *window sequence* (WS) to be the set of polygonal paths from s to g consisting of segments that join points on the edges. A path is said to be *in the interior* of the WS if each common endpoint for two consecutive segments belongs to the relative interior of the edge in question.

The following lemma is essentially the *Fact* given in the proof of Lemma 3.6 in [41], p. 30; however the proof of the *Fact* given in [41] is not correct. The functions corresponding to the length of a line segment in the situation above were claimed to be strictly convex as long as the conditions in our definition of WS above are fulfilled. This is clearly not the case as seen above. However, the *Fact* is true, and we give a proof for our version of it below.

Lemma 4.8. *Let γ be a polygonal path in the interior of a WS defined by the edges $e_1, \dots, e_k$. Let s_i be the coordinate for the intersection point of e_i and γ, as above. Then $L_w(\gamma) = f(s_1, \dots, s_k)$ is a strictly convex function of $s_1, \dots, s_k$.*

Proof: We note that

$$L_w(\gamma) = \sum_{i=0}^{k} w_{m_i} u_i,$$

where u_i is the Euclidean length of the segment of γ from e_i to e_{i+1} for $i = 1, \dots, k-1$ and the Euclidean length of the segment on γ from s to e_1 for $i = 0$ and from e_k to g for $i = k$. Further, w_{m_i} are the corresponding weights. For $i = 1, \dots, k-1$ we consider u_i as function of s_i and s_{i+1} as above, u_0 as a function of s_1, and u_k as a function of s_k. Obviously $L_w(\gamma) = f(s_1, \dots, s_k)$ is a convex function of $s_1, \dots, s_k$. Note that u_0 and u_k are strictly convex functions of s_1 and s_k, respectively. Now we claim that f is a strictly convex function of $s_1, \dots, s_k$.

Let $(s_1^\star, \dots, s_k^\star)$ and $(\overline{s}_1, \dots, \overline{s}_k)$ be two arbitrary points in $(0, L(e_1)) \times \dots \times (0, L(e_k))$. Interpolate linearly between these points. Take an arbitrary Θ, $0 < \Theta < 1$. We claim that

$$f((1-\Theta)(s_1^\star, \dots, s_k^\star) + \Theta(\overline{s}_1, \dots, \overline{s}_k)) < (1-\Theta) f(s_1^\star, \dots, s_k^\star) + \Theta f(\overline{s}_1, \dots, \overline{s}_k).$$

If that is not the case, then

$$\varphi(t) = f((1-t)(s_1^\star, \dots, s_k^\star) + t(\overline{s}_1, \dots, \overline{s}_k))$$

must be linear for $0 \le t \le 1$, since f is convex. Moreover, each

$$\varphi_i(t) = u_i((1-t)(s_i^\star, s_{i+1}^\star) + t(\overline{s}_i, \overline{s}_{i+1}))$$

must also be linear for $0 \le t \le 1$. Let $s_j^\star \ne \overline{s}_j$, and $s_i = (1-t)s_i^\star + t\overline{s}_i$ for each i. Then, according to the above analysis, s_{j+1} and s_{j-1} vary with t. By induction we conclude that all component vary with t, but u_1 and u_k are strictly convex functions of s_1 and s_k respectively. The claim follows by contradiction. □

Remark: The lemma is true even if the assumption (3) is not fulfilled once[4] or one of the assumptions (4) or (5) is not fulfilled. We do not necessarily get uniqueness if we release two or more assumptions as can be seen in Figure 2.

We have the following result:

Theorem 4.9. *Among paths in the interior of a WS there is at most one shortest path. Such a path must satisfy Snell's law at each intersection with an edge.*

Remark: Besides paths of this type one must consider paths passing through corners of $\partial\Omega$ and end-points of edges. We have no systematic way of checking these paths. Identification of locally optimal paths as suggested by Mitchell [41] might be helpful.

In Section 4.6.3 we will analyze a particular case where this result is useful.

The WS:s play, in some sense, the same role for the weighted region problem as homotopy classes do in the non-weighted case. The latter is described by Aronsson in [5], where also more properties for the distance function in the non-weighted case are derived and a nice local structure theorem is presented. So far, we have not been able to generalize any significant portion of these results to the weighted case, which appears to be much harder.

[4] I.e. for one pair of consecutive edges.

4.6. Properties of the distance function

Definition 10. For a fixed source point s, called the injection point, we define the *distance function,* $d(x)$, to be the function mapping $x \in \overline{\Omega}$ to $L_w(\gamma^\star)$ with $g = x$.

4.6.1. Differentiability

Since d is a Lipschitz function we know from Rademacher's theorem [52], Theorem 2.2.1, p. 50, that it is differentiable almost everywhere. The following theorem states sufficient conditions for d not being differentiable:

Proposition 4.10. *If w is not continuous at x then d is not differentiable at x.*

Proof: We place the origin O at x and consider a disc D centered at O with radius $r > 0$ such that for all i, $P_i \cap D$ is either empty or a circle sector. Then we get $\Omega \cap D$ subdivided into N sectors such that w is constant in the interior of each sector. If O is a vertex on some polygon in the subdivision it is possible that $N > 2$; otherwise $N = 2$. Let the weights in the sectors be $w_1, w_2, \ldots, w_N$, ordered such that

$$w_1 \le w_2 \le \ldots \le w_N \text{ and } w_1 < w_N.$$

If d is differentiable at O then $d(y) = d(O) + y \cdot \nabla d(O) + o(|y|)$. First, we claim that $|\nabla d(O)| \le w_1$. To see that, fix a number $k \ge 2$ only depending on the sector angle, such that there exists a disc with center $r/2$ from O and radius r/k contained in the sector with weight w_1; call it sector #1. We can, for each $0 < r' \le r$, find y, y' in sector #1 such that $|y| = r'/2$, $|y - y'| = r'/k$, the disc with radius r'/k and center y is contained in sector #1 and $y - y'$ is parallel to $\nabla d(O)$; see Figure 5. Now we can note that $|d(y) - d(y')| \le w_1 |y - y'|$ since the line segment from y to y' is contained in sector #1. We also get that, for any $\epsilon > 0$,

$$\begin{aligned} |d(y) - d(y')| &= |(y - y') \cdot \nabla d(O) + o(r')| \\ &\ge |y - y'||\nabla d(O)| - |o(r')| \\ &\ge (1 - \epsilon)|y - y'||\nabla d(O)|, \end{aligned}$$

if r' is small enough. This yields $(1 - \epsilon)|\nabla d(O)| \le w_1$ and since ϵ is arbitrary we get $|\nabla d(O)| \le w_1$. Now fix a number $k' \ge 2$, also only depending on sector angle, such that there exists a disc with center z at distance $r/2$ from O and radius r/k' that is contained in the sector with weight w_N; call it sector #N. Let z' be the first point with $|z - z'| = r/k'$ on a shortest path from z to s. Now we have

$$|d(z) - d(z')| \le |\nabla d(O)||z - z'| + |o(r)| \le w_1 |z - z'| + |o(r)| = w_1 r/k' + |o(r)|,$$

but on the other hand we have $d(z) - d(z') = w_N r/k'$, which leads to a contradiction and the theorem follows. $\square$

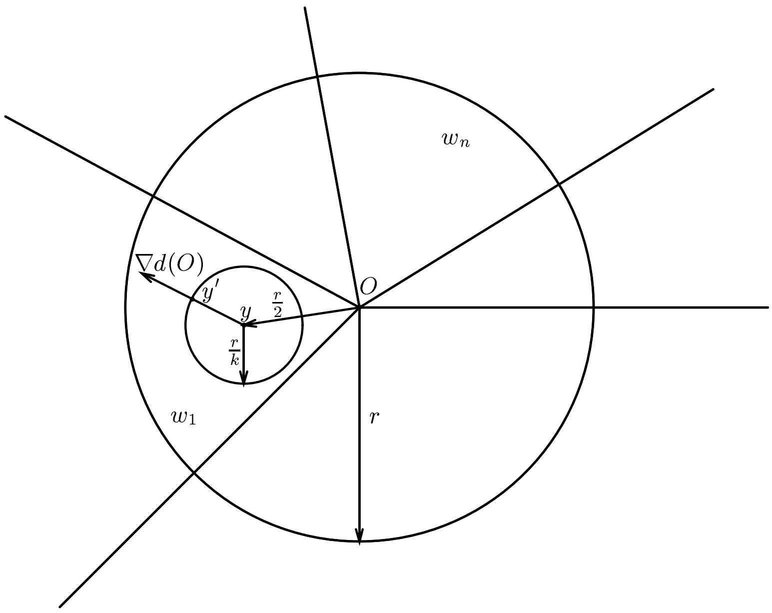


Figure 5: The situation in the proof of Theorem 4.10

Observation 4.11. *If there are two shortest paths γ_1, γ_2 from s to x such that they head off at different directions from x towards s then d is not differentiable at x.*

Proof: The theorem follows directly from the fact that if d is differentiable at x then $\nabla d(x)$ must have the same direction as the last line segment on a shortest path from s to x. □

4.6.2. Weld lines

Definition 11. The *arrival angle* at x of a shortest path for which the last line segment goes from z to x is $\arg(x - z)$, where $\arg(v)$ is chosen in the interval $(-\pi, \pi]$.

Definition 12. A point x is situated on a *weld line* if there exist at least two shortest paths to x with different arrival angles at x.

The physical interpretation of a point on a weld line is that there are at least two flow-fronts meeting there.

4.6.3. Structure of the distance function in a particular case

In this section we will analyze the structure of the distance function in a specific geometry, namely the example from Boshouwers and van der Werf [10] described in Figure 29 in Section 7.4 in this work.

Due to symmetry it is enough to study the domain for $x \geq 0$. We will further subdivide this domain into 4 regions to be analyzed separately. These 4 regions are

$$\begin{aligned}
\Omega_1 &= \{(x,y) \in \mathbb{R}^2 : 0 \leq x \leq 20, 0 \leq y \leq 15\} \\
&\quad \cup \{(x,y) \in \mathbb{R}^2 : 15x - 20y \geq 0, x \leq 50, y \geq 0\} \\
\Omega_2 &= \{(x,y) \in \mathbb{R}^2 : 15x - 20y \leq 0, x \geq 20, y \leq 60\} \\
\Omega_3 &= \{(x,y) \in \mathbb{R}^2 : 0 \leq x \leq 20, 45 \leq y \leq 60\} \\
\Omega_4 &= \{(x,y) \in \mathbb{R}^2 : 0 \leq x \leq 20, 15 \leq y \leq 45\}
\end{aligned}$$

and we denote the edges

$$\begin{aligned}
e_1 &= \Omega_4 \cap \Omega_1 \\
e_2 &= \Omega_4 \cap \Omega_2 \\
e_3 &= \Omega_4 \cap \Omega_3
\end{aligned}$$

and the vertices $v_1 = (20, 15)$, $v_2 = (20, 45)$, and $v_3 = (0, 45)$, see Figure 6.

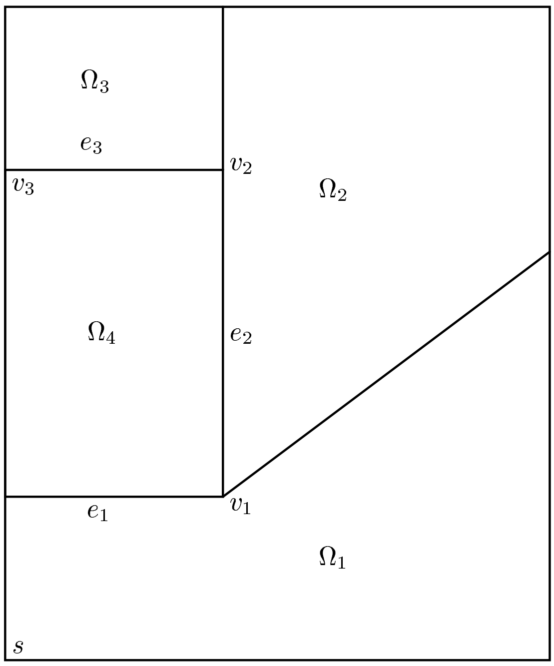


Figure 6: The subdivision of the domain

We will rescale the weight so that the weight function function w is 1 in $\Omega_1 \cup \Omega_2 \cup \Omega_3$ and 2 otherwise. We have the injection point as starting point $s = (0, 0)$. It is clear that the shortest path from s to any point $g \in \Omega_1$ is the straight line segment $\overrightarrow{sg}$ and we have that

$$d|_{\Omega_1}(x,y) = \sqrt{x^2+y^2}.$$

It has been observed by Rowe and Richbourg in [45], PM 9 p. 59, that "corner cutting" does not occur if the ratio of the weights is less than the sine of half the angle in the expensive region at the corner. This is the case since the greatest advantage of a shortcut occurs when it forms an isosceles triangle with the corner vertex. In our case we have $\frac{1}{2} < \sin(\frac{1}{4}\pi)$ and consequently a shortest path does not cross two successive edges of Ω_4. Thus the shortest path from s to $g = v_2$ must be $\overrightarrow{sv_1}$ followed by $\overrightarrow{v_1v_2}$, since a path from s to v_2 connecting $(e_1,\, e_3)$ is obviously longer. Now we know that the shortest path to all points on e_2 passes through v_1.

Since we have no corner cutting there are only two possible candidates for shortest paths to $g = v_3$. The first candidate γ_1 is $\overrightarrow{sv_1}$ followed by $\overrightarrow{v_1v_2}$ and $\overrightarrow{v_2v_3}$ and the second γ_2 is $\overrightarrow{sv_3}$. We have that $L_w(\gamma_1) = 25+30+20 = 75$ and $L_w(\gamma_2) = 15 + 2\cdot 30 = 75$ so in fact both are shortest paths! However the shortest path to all other points on e_3 passes through v_1 and v_2 only. This is clear since the path to a fixed point on e_3, passing only through the edge e_1, has greater weighted length than $d(v_3)$ and on the other hand the path to the fixed point through v_1 and v_2 has less weighted length than $d(v_3)$.

Now it is clear that the shortest path to a point $g \in \Omega_2$ is $\overrightarrow{sv_1}$ followed by $\overrightarrow{v_1g}$ since entering Ω_4 is impossible (shortest paths to different points can not cross each other) and we have

$$d|_{\Omega_2}(x,y) = \sqrt{(x-20)^2+(y-15)^2} + 25.$$

Similarly the shortest path to a point $g \in \Omega_3$ is $\overrightarrow{sv_1}$ followed by $\overrightarrow{v_1v_2}$ and $\overrightarrow{v_2g}$. This is the case since a path to g that crosses e_3 at a point y has greater weighted length than the path through g, y, v_2, v_1, s according to the above analysis, and clearly the path g, v_2, v_1, s is shorter. We have

$$d|_{\Omega_3}(x,y) = \sqrt{(x-20)^2+(y-45)^2} + 55.$$

For the shortest path from s to a point $g = (x,y) \in \Omega_4$ we have to consider three candidates. The first candidate is the path from s to g connecting (e_1), i.e. crossing e_1 at a point $p_1 = (x_1, 15)$. We get that

$$d_1(x,y) = \min_{0\le t\le 20} \sqrt{t^2+15^2} + 2\sqrt{(x-t)^2+(y-15)^2}. \tag{4.14}$$

The path obeys Snell's law at p_1. The second candidate is $\overrightarrow{sv_1}$ followed by $\overrightarrow{v_1p_2}$ and $\overrightarrow{p_2g}$, where $p_2 = (20, y_2) \in e_2$. The path has to obey Snell's law at p_2 so if $y\sqrt{3} + x \ge 20 + 15\sqrt{3}$ then

$$(y-y_2)\sqrt{3} = 20 - x \text{ and thus } y_2 = y + \frac{x-20}{\sqrt{3}},$$

and
$$d_2(x,y) = 25 + (20-x)\sqrt{3} + y - 15 = (20-x)\sqrt{3} + y + 10.$$

Otherwise we have that d_2 is greater than the weighted length of the path through y_2 since the path will exit through e_1 and in d_2 be charged with the weight 2 until the x-coordinate is 20. Further d_1 is less than (or equal if $(x,y) = (20,15)$) to the length of the shortest path through e_2 since the shortest path through e_2 obviously goes through v_1 and that path can be shortened by moving that vertex to the left.

Finally, the third candidate is $\overrightarrow{sv_1}$ followed by $\overrightarrow{v_1v_2}, \overrightarrow{v_2p_3}$ and $\overrightarrow{p_3g}$ where $p_3 = (x_3, 45) \in e_3$. The path obeys Snell's law at p_3 so if $x\sqrt{3} - y \leq 45 + 20\sqrt{3}$ then
$$(x_3 - x)\sqrt{3} = 45 - y \text{ and thus } x_3 = x + \frac{45-y}{\sqrt{3}},$$

and
$$d_3(x,y) = 55 + 20 - x_3 + 2\sqrt{(x_3-x)^2 + (45-y)^2} = 75 - x + (45-y)\sqrt{3}.$$

Otherwise the situation is similar to the one above for d_2, here d_3 is greater than the weighted length of the path through x_3 and that path has larger weighted length than the shortest through e_3 (which goes through v_2) which can be shortened if it is allowed to cross e_2 and thus d_2 is less than the weighted length of the shortest path through e_3.

We conclude that
$$d|_{\Omega_4}(x,y) = \min\{d_1(x,y), d_2(x,y), d_3(x,y)\}.$$

The distance function is plotted in Figure 7.

Now we want to describe the curves where $d_1 = d_2, d_1 = d_3$, and $d_2 = d_3$. We introduce new coordinates. For d_1 we will use
$$\alpha = \arctan\frac{15}{x_1}, \text{ and } u_1 = \sqrt{(x-x_1)^2 + (y-15)^2},$$

where x_1 is, as above, that value of t where the minimum is attained in (4.14). In the new coordinates d_1 takes the expression
$$d_1 = 15\csc\alpha + 2u_1,$$

and the relation between old and new coordinates are
$$x = 15\cot\alpha + \frac{u_1}{2}\cos\alpha, \text{ and } y = 15 + u_1\sqrt{1 - \tfrac{1}{4}\cos^2\alpha} = 15 + \frac{u_1}{2}\sqrt{3 + \sin^2\alpha}.$$

For d_2 we will use the coordinates
$$u_2 = \sqrt{(20-x)^2 + (y-y_2)^2}, \text{ and } v_2 = y_2 - 15.$$

Now d_2 is expressed as
$$d_2 = 25 + v_2 + 2u_2,$$

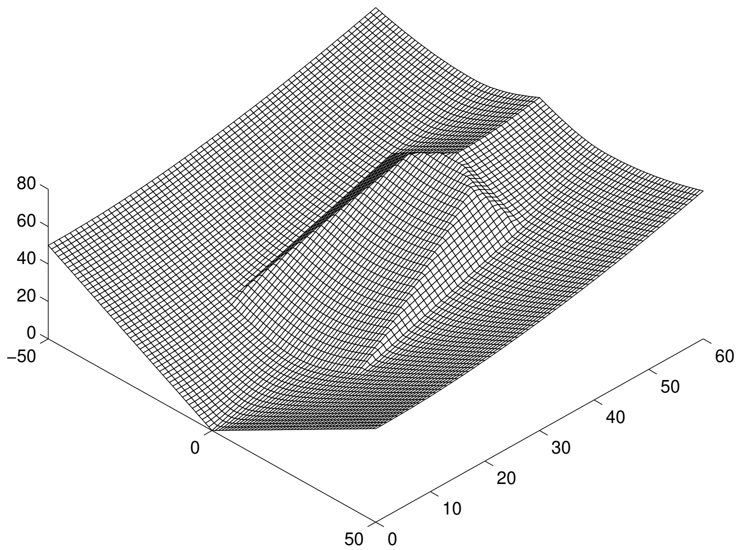


Figure 7: The distance function $d(x, y)$

and we have the relations

$$x = 20 - \frac{\sqrt{3}}{2}u_2, \text{ and } y = 15 + v_2 + \frac{1}{2}u_2.$$

For a fixed α requiring that $d_1 = d_2$ in a point leads to the system of equations

$$\begin{aligned} 15\csc\alpha + 2u_1 &= 25 + v_2 + 2u_2 \\ 15\cot\alpha + \frac{u_1}{2}\cos\alpha &= 20 - \frac{\sqrt{3}}{2}u_2 \\ 15 + u_1\frac{1}{2}\sqrt{3+\sin^2\alpha} &= 15 + v_2 + \frac{1}{2}u_2 \end{aligned}$$

which is linear in u_1, u_2, and v_2. Solving it we obtain

$$u_1 = 10\frac{3 + 3\sqrt{3}\cos\alpha - (5 + 4\sqrt{3})\sin\alpha}{\sqrt{3+\sin^2\alpha}\sin\alpha - (4 + \sqrt{3}\cos\alpha)\sin\alpha}$$

which gives the following expression for the curve $d_1 = d_2$ parameterized by α

$$\begin{aligned} x(\alpha) &= 5\left(\frac{3}{\tan\alpha} + \frac{3 + 3\sqrt{3}\cos\alpha - (5 + 4\sqrt{3})\sin\alpha}{\sqrt{3+\sin^2\alpha}\sin\alpha - (4 + \sqrt{3}\cos\alpha)\sin\alpha}\cos\alpha\right), \\ y(\alpha) &= 5\left(3 + \frac{3 + 3\sqrt{3}\cos\alpha - (5 + 4\sqrt{3})\sin\alpha}{\sqrt{3+\sin^2\alpha}\sin\alpha - (4 + \sqrt{3}\cos\alpha)\sin\alpha}\sqrt{3+\sin^2\alpha}\right), \end{aligned}$$

over some appropriate α-interval. For d_3 we will use the coordinates

$$u_3 = \sqrt{(x - x_3)^2 + (y - 45)^2}, \text{ and } v_3 = 20 - x_3.$$

With these coordinates we express

$$d_3 = 55 + v_3 + 2u_3,$$

and we have the relations

$$x = 20 - v_3 - \frac{1}{2}u_3, \text{ and } y = 45 - \frac{\sqrt{3}}{2}u_3.$$

By again holding α fixed and now requiring $d_1 = d_3$ we have the system of equations

$$\begin{aligned} 15\csc\alpha + 2u_1 &= 55 + v_3 + 2u_3 \\ 15\cot\alpha + \frac{1}{2}\cos\alpha u_1 &= 20 - v_3 - \frac{1}{2}u_3 \\ 15 + \frac{1}{2}u_1\sqrt{3+\sin^2\alpha} &= 45 - \frac{\sqrt{3}}{2}u_3 \end{aligned}$$

which is linear in u_1, u_3, and v_3. By solving the equation system we obtain

$$u_1 = 30\frac{2\sqrt{3} + 5 - \cot\alpha - \csc\alpha}{\sqrt{9+3\sin^2\alpha} + 4 + \cos\alpha}$$

which gives the following expression for the curve $d_1 = d_3$ parameterized by α

$$\begin{aligned} x(\alpha) &= 15\left(\cot\alpha + \frac{2\sqrt{3} + 5 - \cot\alpha - \csc\alpha}{\sqrt{9+3\sin^2\alpha} + 4 + \cos\alpha}\cos\alpha\right), \\ y(\alpha) &= 15\left(1 + \frac{2\sqrt{3} + 5 - \cot\alpha - \csc\alpha}{\sqrt{9+3\sin^2\alpha} + 4 + \cos\alpha}\sqrt{3+\sin^2\alpha}\right), \end{aligned}$$

again over some suitable α-interval. The curve $d_2 = d_3$ is obtained directly from the original expressions for d_2 and d_3, namely

$$\sqrt{3}(20 - x) + y + 10 = \sqrt{3}(45 - y) - x + 75$$

which yields the straight line

$$(\sqrt{3} - 1)x - (\sqrt{3} + 1)y + 5(13 + 5\sqrt{3}) = 0.$$

A plot of these curves can be found in Figure 8.

The curve $d_1 = d_2$ has a unit tangent vector at $(x(\alpha), y(\alpha))$ that we will denote $\overline{e}$. Since $d_1 = d_2$ on the curve we have $\nabla d_1 \cdot \overline{e} = \nabla d_2 \cdot \overline{e}$ on the curve, but we also know that $|\nabla d_1| = |\nabla d_2| = w$ so we obtain that $\nabla d_1 + \nabla d_2$ is parallel to $\overline{e}$. From this we can obtain the angle of the weld line at v_1. Using notation from Figures 9 and 10 we have that

$$\delta = \beta + \gamma = \beta + \frac{1}{2}\left(\frac{\pi}{2} + \alpha_o - \beta\right) = \frac{\pi}{4} + \frac{\alpha_o + \beta}{2}$$

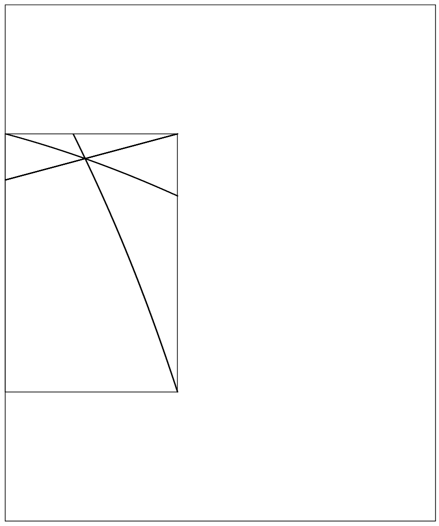


Figure 8: The curves $d_1 = d_2, d_1 = d_3$ and $d_2 = d_3$.

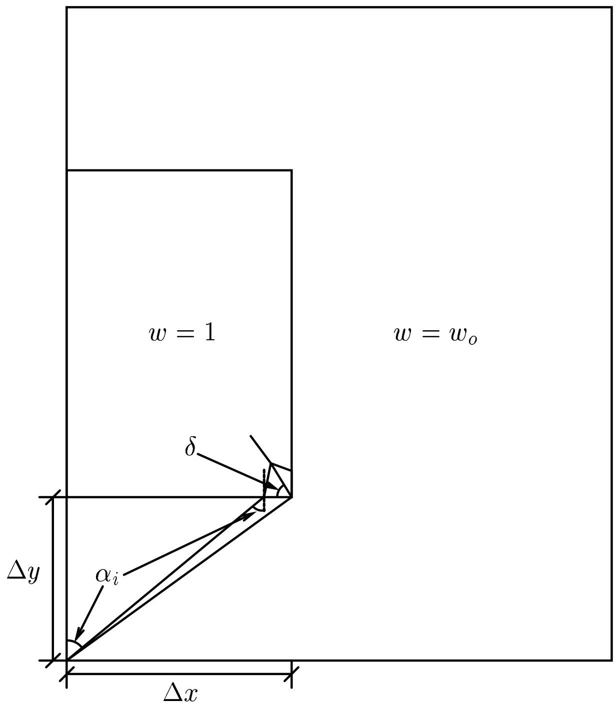


Figure 9: Two shortest paths meeting at a weld line near v_1

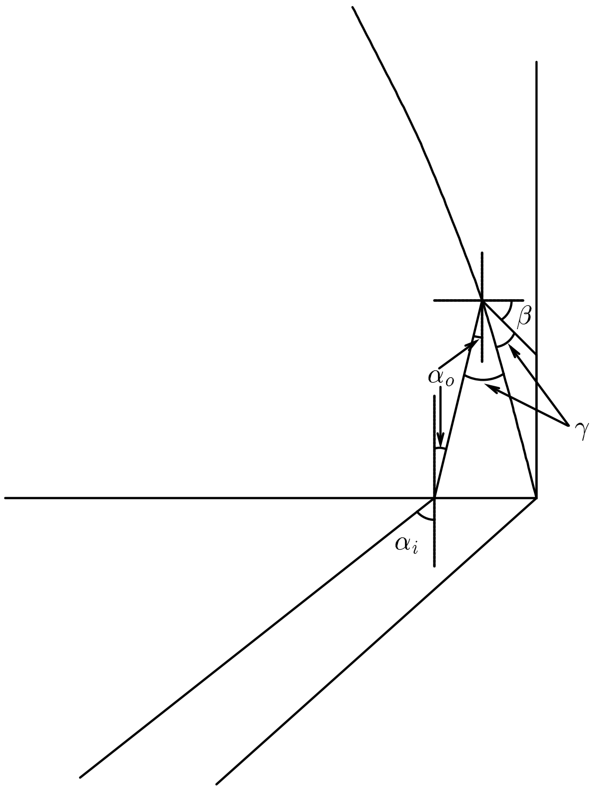


Figure 10: Closeup of the situation in Figure 9.

The paths obey Snell's law which gives

$$\begin{aligned} \sin\alpha_o &= w_o \sin\alpha_i \\ \sin\beta &= w_o \end{aligned}$$

At the corner v_1 we have

$$\alpha_i = \arctan\frac{\Delta x}{\Delta y}.$$

This yields

$$\delta = \frac{\pi}{4} + \frac{\arcsin\left(w_o \frac{\Delta x}{\Delta h}\right) + \arcsin w_o}{2}$$

where $\Delta h = \sqrt{(\Delta x)^2 + (\Delta y)^2}$. In the present case $w = \frac{1}{2}, \Delta x = 20$, and $\Delta y = 15$. Entering that into the expression above gives

$$\delta = \frac{\pi}{3} + \frac{1}{2}\arcsin\frac{2}{5} \approx 71.8°.$$

Naturally, the angles at v_2, v_3, and at the intersection of the curves can be computed in a similar fashion.

5. Computational aspects

We know now that a shortest path is necessarily a polygonal curve having at most C shift points (see Lemma 4.4). The shift points (including corners) can only occur on separation lines between "sub-polygons" and vertices of $\overline{\Omega}$. By discretizing these separation lines and only considering polygonal curves having shift points restricted to the discretization points, we obtain a smaller class of polygonal curves. For a fine discretization this smaller class has sufficient approximation properties.

The strategy of our computations is therefore to

(a) choose a convenient discretization, i.e. the set A below

(b) compute the approximate distance function $d_h(x)$ for all $x \in A$

(c) compute $d_h(x)$ for all grid points x on a suitably chosen grid.

Although steps (a) and (c) are of some interest (and optimal choices not obvious) the subsequent discussion will essentially be focused on step (b).

5.1. Approximation of $\gamma^\star$

We introduce a discretization of the boundaries of the polygons in the subdivision of $\overline{\Omega}$. For $h > 0$ let

$$A = \{a_i\}_{i=1}^N \subset \bigcup_{j=1}^n \partial P_j$$

be such that

1. each vertex of a polygon P_j belongs to A, and
2. each separation line between polygons P_i and P_k is divided by A into segments of length $< h$.

Our aim is to get an error estimate comparing a true shortest path and an approximating path obtained by restricting the shift points to A.

Theorem 5.1. *Let $\gamma \in M$ be a curve obtained by reduction as in Lemma* 4.4 *and thus having at most C shift points. Then there exists a polygonal curve $\tilde{\gamma} \in M$, having shift points in A only, and such that $L_w(\tilde{\gamma}) - L_w(\gamma) \leq Chw_n$.*

Proof: Consider $f(t) = p(x(t))$ as in Observation 4.1. Since γ is reduced as in Lemma 4.4 the case 2(b)ii in Observation 4.1 can not occur. Decompose γ into line segments between points corresponding to 2a or 2(b)iii in Observation 4.1. Let $\gamma_0 = \gamma$ and r be the number of shift points for γ not in A. Now we do as the follows recursively for $i = 1, \dots, r$ (shift points not in A denoted $\{x_i\}_{i=1}^r$.) We obtain γ_i from γ_{i-1} by replacing the two line segments adjacent to x_i by the line segments obtained by moving x_i to a point in A on the same edge as x_i and at a distance at most $\frac{1}{2}h$ from x_i. By the triangle inequality the new line segments are at most $\frac{1}{2}h$ longer than the old ones. Since the polygons are convex and the line segments in the decomposition are contained in one polygon each, the new line segments are contained in the same polygons. Thus the new line segments are charged with the same weight (which is at most w_n). Let $\tilde{\gamma} = \gamma_r$; then we get $L_w(\tilde{\gamma}) - L_w(\gamma) < Chw_n$. $\square$

For $\gamma = \gamma^\star$ the theorem says that there exists a polygonal curve $\tilde{\gamma}$, through points in A such that

$$L_w(\tilde{\gamma}) - L_w(\gamma^\star) \le Chw_n = \left(\prod_{0<i\le k} (n_i + 1) - 2 \right) hw_n. \tag{5.15}$$

We believe that this estimate is very rough for most geometries. We now restrict our attention to polygonal curves having shift points in A only.

5.2. An algorithm for computing $d_h(x)$

Definition 13. For a fixed source point s, called the injection point, and a discretization of the edges as in Section 5.1 we define $d_h(x)$ to be the distance function for polygonal curves having shift points in A only.

Note that $d_h(x)$ depends only on the discretization itself. The error estimate (5.15) is expressed through the parameter h only.

We shall give an algorithm that computes $d_h(x)$ for each point x in a uniform grid

$$G = \{(i\Delta, j\Delta)\}_{i,j\in\mathbb{Z}} \cap \overline{\Omega},$$

on the weighted region. Note that

$$\#G \le \left\lfloor \frac{\max\limits_{(x_1,x_2)\in\overline{\Omega}} x_1 - \min\limits_{(x_1,x_2)\in\overline{\Omega}} x_1}{\Delta} + 1 \right\rfloor \left\lfloor \frac{\max\limits_{(x_1,x_2)\in\overline{\Omega}} x_2 - \min\limits_{(x_1,x_2)\in\overline{\Omega}} x_2}{\Delta} + 1 \right\rfloor$$

The main structure of our algorithm is as follows:

The weighted distance is computed for all discretization nodes, one at a time, in order of increasing distance. (Finding the distance to an arbitrary point in $\overline{\Omega}$ is then easy.) The discretization nodes are divided into two sets: R_i (=reached), for which the distance has been computed and N_i, for which the distance is not yet known. At each step of the algorithm one node $y_i \in N_i$ (the closest to s, or one of the closest) is transferred to R_i, so that $R_{i+1} = R_i \cup \{y_i\}$. For improved computational economy the algorithm keeps a tentative distance $t_i(x)$ (=an upper bound for $d_h(x)$) for each $x \in N_i$. This bound is successively updated.

The algorithm can be described by its INPUT, OUTPUT and three stages, BUILD, PRECOMP and COMPUTE.

INPUT Resolution of the computational grid G. The region with its subdivision and weights. Discretization parameter for each edge. The injection point.

OUTPUT For each point in the grid G, the distance function $d_h(x)$ and the arrival angle at x for the approximating shortest path found by the algorithm.

BUILD Build the data structures used by the algorithm.

PRECOMP Determine the distance function on the discretization points as a precomputation step.

COMPUTE For each point in the grid G compute the approximate distance function $d_h(x)$ and the arrival angle at x for $\hat{\gamma}$.

We shall in the following subsections describe each stage and the input in more detail.

5.2.1. INPUT

1. The resolution Δ of the grid G.
2. Coordinates of the injection point s.
3. Coordinates for all edge endpoints $\{p_i\}$ in the subdivision of the weighted region.
4. For each edge e_j in the subdivision
   (a) the endpoints as a pair of point indices (f_j, t_j), where the edge endpoints are p_{f_j} and p_{t_j},
   (b) a discretization parameter, h_j.
5. For each polygon P_l in the subdivision
   (a) the weight w_l corresponding to the unit cost of travel within P_l,
   (b) a list of the edges (with orientation) $\{\pm e_j^l\}$ on the polygon boundary in clockwise order.

5.2.2. BUILD

1. Build a list of all polygons in the subdivision of the weighted region.
2. Build a list of all edges in the subdivision of the weighted region.
3. For each polygon,
   (a) build a list of all edges on the polygon boundary,
   (b) build a list of all vertices on the polygon,

   both in clockwise order.

4. For each edge e_j, build a list of discretization points, $\{a_q^j\}_{q=0}^{Q}$, with
$$a_q^j = p_{f_j} + qh_j \frac{p_{t_j} - p_{f_j}}{L(p_{t_j} - p_{f_j})}, \quad 0 \le q \le Q - 1 < \frac{L(p_{t_j} - p_{f_j})}{h_j} \le Q$$
and $a_Q^j = p_{t_j}$.

5. For each polygon determine the set A_l of discretization points on the polygon's boundary and if $s \in P_l$ then also include s in A_l.

Note that
$$A \cup \{s\} = \bigcup_{l=1}^{n} A_l \text{ and } h = \max_j h_j$$
where A and h were introduced in Section 5.1.

5.2.3. PRECOMP

Consider a graph having the injection point s and all discretization points as nodes and arcs between each pair of points in the same polygon in the subdivision of $\overline{\Omega}$. Note that the requirement (1) on the discretization in Section 5.1 guarantees that all vertices of $\overline{\Omega}$ are included among the discretization points. Further note that a discretization point in Ω belongs to at least two polygons, more than two if it is a vertex. Let each arc be assigned a cost equal to the corresponding weighted distance restricted to the polygon. Now we are going to construct a shortest path tree for the graph, using Dijkstra's algorithm, originally described by Dijkstra in [13]. A discussion of Dijkstra's algorithm and its variants can be found in e.g. Gibbons [22], p. 13 and p. 33, or Evans and Minieka [19], Ch. 4. When we have the shortest path tree, we also know $d_h(x)$ for all discretization points.

The basic structure of the algorithm is as follows:

1. A shortest discretized path from y to s must pass through supporting nodes x in order of strictly decreasing $d_h(x)$.

2. The algorithm guarantees by construction that $d_h(y) \ge d_h(x)$ for all $y \in N_i$ and $x \in R_i$.

3. Now, knowing $d_h(x)$ for all $x \in R_i$, one can compute $d_h(y)$ for at least one $y \in N_i$, and thus transfer it to R_{i+1}.

We now describe Dijkstra's algorithm in our context in more detail. In the following we will write distance for the approximating weighted distance, d_h, and "closest to s" will be understood in the sense of least approximating weighted distance.

The basic idea is to partition the discretization nodes into two sets, called the reached points R_i and the not yet reached points N_i, such that R_i contains i points and for each point in R_i the distance is less than or equal to the distance for an arbitrary point in N_i. In our application this means that at a time corresponding to i the polymer fluid has reached the points in R_i. This set R_i is not necessarily uniquely determined, since several points can have the same distance. For a description of a connection between distance and the fluid front propagation see Section 8.1.

The main step is as follows: Let y_i be a point in N_i that is closest to s (among the points in N_i). Now we can construct R_{i+1} by adding y_i to R_i and N_{i+1} by removing y_i from N_i.

To find y_i in a computationally economic way we keep a tentative distance $t_i(x)$ for each node $x \in N_i$. This tentative distance has the property that $\min_{x \in N_i} t_i(x) = t_i(y_i) = d_h(y_i)$. We obtain this by letting $t_i(x)$ be the length of the shortest of the paths from s to x that consists of the path to a reached point z on the same polygon as x followed by the line segment connecting z with x. If there are no such paths we let $t_i(x) = \infty$.

We can observe that if $x \in N_i$ and x is not on the same polygon as y_i then $t_{i+1}(x) = t_i(x)$. This means that we can update $t_i(x)$ by only looking at the not yet reached points on the same polygon as y_i.

We formally state Dijkstra's algorithm as:

1. Let $y_1 = s, d_h(s) = 0, R_1 = \{s\}, N_1 = A$.
2. For each $x \in N_1$ let
$$t_1(x) = \min\{w_l|x - s| : x \in A_l \text{ and } s \in A_l\},$$
where $\min \emptyset = \infty$.
3. For $i = 1, 2, \dots, \#A$ recursively repeat the steps 3a–3c below
   (a) Let y_i be a point such that $t_i(y_i) = \min\limits_{x \in N_i} t_i(x)$.
   (b) Let $R_{i+1} = R_i \cup \{y_i\}, N_{i+1} = N_i \setminus \{y_i\}, d_h(y_i) = t_i(y_i)$.
   (c) For each $x \in N_{i+1}$ let
   $$t_{i+1}(x) = \min(t_i(x), \min\{d_h(y_i) + w_l|x - y_i| : x \in A_l \text{ and } y_i \in A_l\}),$$
   where $\min \emptyset = \infty$ as above.

Note that in the minimizations over the sets above the points are fixed and the parameter l varies, i.e. the minimization is over all l such that the polygon P_l contains both points. The minimum is obtained for the polygon with least weight, which coincides with the polygon with least index l because of our ordering. Thus the minimization is actually never performed, instead the polygons are searched in order of increasing l until a polygon that contains both points is found.

In an implementation of the algorithm y_i would be the value of a variable y at repetition i of the loop 3a–3c. Similarly for R_i and N_i that could be implemented as two lists of points. We can let each point x have a variable that contains $t_i(x)$ when the node is not yet reached and $d_h(x)$ when the point is reached. Then the equation $d_h(y_i) = t_i(y_i)$ is a matter of definition and not a step actually performed by the computer. We can reduce the need of lists to one if we let each point have a variable indicating whether it is reached or not. Then updating R_i and N_i only consists of changing the value of that variable for the point y_i instead of removing an element from the list N_i and adding an element to the list R_i. The step 3c is implemented as traversing all polygons P_l that contain y_i and for all points in A_l (A_l is defined in Section 5.2.2 item 5) checking if the point is not yet reached and then updating $t_i(x)$ according to the description above.

5.2.4. COMPUTE

For each point x in the grid G evaluate the polygon reference function $p(x)$ as defined in Definition 4. We get a polygon P_l with least weight among those containing the point. Now we have

$$d_h(x) = \min_{y \in A_l} \{w_l|x-y| + d_h(y)\} = w_l|x-z| + d_h(z)$$

where $d_h(y)$ is already computed in the PRECOMP stage and A_l is defined in the BUILD stage, see Section 5.2.2 item 5. The observant reader may note that d_h is not necessarily continuous at separation lines and the above description of d_h does not exactly correspond to Definition 13, but the deviation is not important, the error is $O(h)$, and it is possible to modify either the definition or the algorithm to have exact correspondence; however, trying to maximize simplicity, we have not done that.

5.3. Artificial separation lines

The subdivision into convex polygons necessitates the use of artificial separation lines, i.e. edges between polygons with the same weight. If an optimal path is crossing an artificial separation line we might introduce a corner on the approximate shortest path at the artificial separation line. The error introduced is by equation 5.15 in Section 5.1 of order $O(h)$. In absolute terms we get a smaller error if the artificial separation lines are as few as possible, but this does not change the order of the error. When subdividing the domain into convex polygons it may happen that one needs a lot of polygons in the subdivision. Consider e.g. the situation in Figure 11. Each corner on the regular octagon constitutes a re-entrant corner unless we add an edge. However, we do not need to add vertices as done in Figure 11, the existing corners on the bounding rectangle could be enough. If we wish to approximate a circular region with a n-gon this gives rise to n artificial separation lines.

5.4. Complexity of computations

As for complexity, our algorithm is not directly comparable to the algorithms mentioned in Section 4.3, since those have complexity measured in terms of

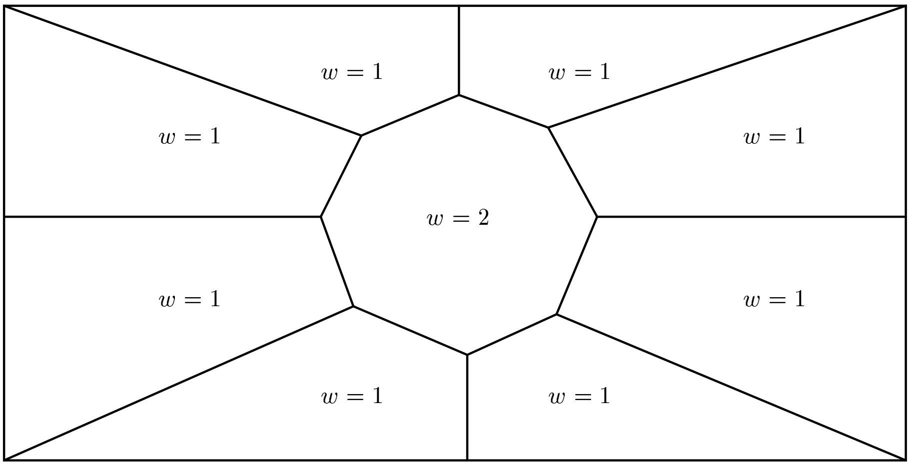


Figure 11: Example where many polygons are needed in the subdivision of Ω.

the number of edges or vertices of the polygons in the subdivision, while we will give our complexity in terms of the size of the graph on which we run Dijkstra's algorithm, i.e. the number of discretization nodes, and the number M of points in the computational grid G introduced in Section 5.2. The complexity is coupled to the accuracy in an immediate way, since the discretization error is $O(h)$ and the number N of discretization nodes is $\Omega(\frac{1}{h})$. The BUILD stage, see Section 5.2.2, is trivially running in $O(N)$ time. The PRECOMP stage, see Section 5.2.3, is a straightforward implementation of Dijkstra's algorithm on a graph with N nodes and thus runs in $O(N^2)$ time. Finally the COMPUTE stage, see Section 5.2.4, runs in time $O(NM)$ since we for each point will first search among at most n polygons, where $n < N$, and then minimize over at most N nodes. By recording the most recent polygon and checking if the new grid point is well inside this polygon we will almost always only check one polygon and thus reduce the constant in the complexity $O(NM)$. By using a "tolerance" δ for numerical stability we obtain a small neighbourhood of the boundary where we check against all other polygons, as well as outside[5] the "current" polygon. This tolerance, δ, can indeed be much smaller than the distance between neighbouring grid points, Δ. Nevertheless, in practice for the geometries the author has studied the COMPUTE stage dominates the execution time except for cases of extremely fine discretization, or many edges as when approximating circular regions, see Section 7.

[5] With respect to the tolerance.

6. How to include boundary influence

It has been observed by Aigner in [2] that the distance model tends to overestimate the flow through narrow channels. It is obvious that the Hele–Shaw assumptions are not satisfied in a very narrow channel. In this section we will discuss an ad-hoc method that takes such effects into account without sacrificing the formulation of the problem as a WRP.

We will do this in two cases. First we study the reduction of flow near the boundary of the domain. Then we compute the flow through a narrow channel. Power-law fluid behaviour is assumed.

6.1. Derivation

The traditional boundary condition at the outer boundary Γ_1 for Hele–Shaw flow, also used in the main part of this work, is just $\frac{\partial p}{\partial n} = 0$, expressing the fact that the flow cannot cross the outer boundary. This is, so to say, the condition on the macro scale. On the micro scale, however, one *has* the no-slip condition, but it is considered that the influence from the outer boundary Γ_1 is negligible at distance more than, let us say, $3h$. In view of the big aspect ratio of lateral dimensions to thickness h, one simply neglects the thin boundary layer that must exist along Γ_1 and thus only the condition $\frac{\partial p}{\partial n} = 0$ remains. This, of course, facilitates the treatment enormously, but also becomes questionable in cases where the aspect ratio is not big enough.

In practice, injection moulds often have parts or sections where the aspect ratio is not big enough to ensure the usefulness of the Hele–Shaw approach. What can then be done? We will sketch an approach here, and numerical examples are given in Section 7.

The idea of our approach is to consider the flow near Γ_1 as a *pipe flow along* Γ_1. We then use the "full" no-slip condition, and the flow velocity at various points in the fictitious pipe is found by numerical solution of a two-dimensional boundary value problem, assuming the pressure gradient to be known and directed along the pipe. The boundary value problem in question is elliptic and non-linear (except in the Newtonian case).

Fortunately, there are good finite element codes available for the numerical solution, some of them developed in Linköping. The boundary value problem is posed on an oblong rectangle, and the transition back to the moulding problem is simply made by taking averages of the flow velocity across the thickness, i.e. normally over the vertical coordinate. Then the mould thickness h is replaced by a reduced, "equivalent thickness" near Γ_1. It is important for this approach that the non-linear proportionality (i.e. the exponent) between flow velocity and pressure gradient is the same for the pipe flow and the Hele–Shaw flow. This exponent is also independent of the rectangle, i.e. the cross-section of the "pipe". Let us take a look at this:

For the polymer flow in the mould we have from equation (2.8) that the polymer flow, for any value of the thickness $F(x, y)$, is directly proportional to $|\nabla p|^{\frac{1}{n}}$, i.e. *the pressure gradient, raised to the power* $\frac{1}{n}$.

Now for an approximating pipe flow, we know from [4] that the flow velocity u in the pipe is found from a Dirichlet problem (over the pipe's cross-section) for the non-linear elliptic PDE

$$\operatorname{div}\left(|\nabla u|^{n-1}\nabla u\right) = C|\nabla p| \tag{6.16}$$

where $|\nabla p|$ = pressure gradient and the constant C is unimportant for the moment. Note that the boundary condition is $u = 0$. Clearly, then $|\nabla u|^n \propto |\nabla p|$, i.e. *the flow velocity is proportional to* $|\nabla p|^{\frac{1}{n}}$. Note that equation (6.16) is the q-Poisson equation and in our case $n > 0$, i.e. $q = n + 1 > 1$, and we have the standard mathematical case.

It is therefore possible to take into account boundary influence within the distance model by using an appropriate weighted region formulation. One simply introduces a few "expensive" subregions near the boundary Γ_1. Further details will be clear from the examples. It should be noted here that the reduced thickness depends on the power-law exponent!

6.2. Boundary influence as a WRP

6.2.1. Flow reduction near the boundary

To study the flow reduction near the boundary we have computed the stationary flow through a pipe with a rectangular cross-section of height 1 and length 16 using the program PLTMG (Piecewise Linear Triangular finite elements and MultiGrid iterative procedure). The programming was done by Klas Samuelsson using a finite element method with mesh refinement for solving equation (6.16) on a quarter of the domain (due to symmetry) with boundary condition $u = 0$ on the walls and $\frac{\partial u}{\partial n} = 0$ on the symmetry lines. According to [4], p. 137, we have $|\nabla p| = M$ =constant. The result is plotted in Figure 12, where we plot

$$x \mapsto \frac{\int_0^1 u(x,z)\,dz}{\int_0^1 u_a(z)\,dz}$$

where $u\vec{e}_y$ is the computed velocity field in the cross-section, $u_a\vec{e}_y$ is the asymptotical velocity field i.e. for $x \to \infty$, x is the distance from the wall, and $\vec{e}_y$ is the flow direction which is perpendicular to the cross-section of the pipe.

To enter this as a WRP we construct e.g. four regions:

A for $0 < x < \frac{1}{2}h$

B for $\frac{1}{2}h < x < h$

C for $h < x < 2h$

D for $2h < x < 3h$

where x is the distance to the boundary. In each region the weight will be $\frac{1}{ch}$ where c is obtained from Table 1. The figures in the table are computed by averaging the flow plotted in Figure 12 for each region.

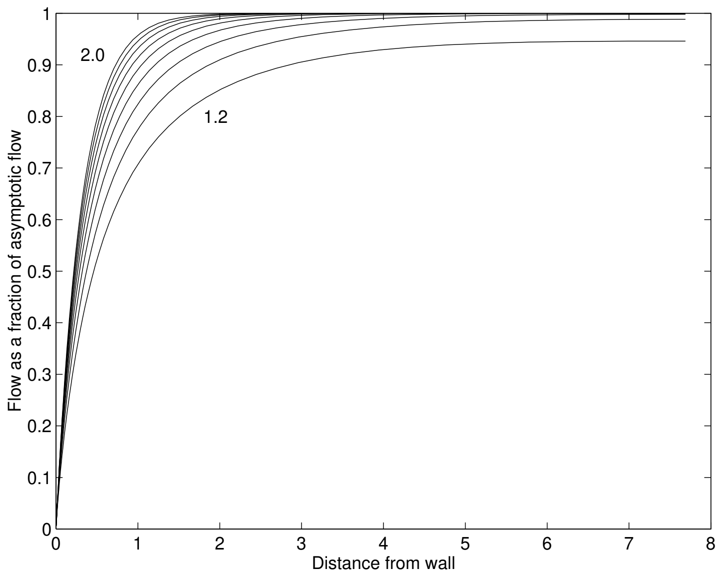


Figure 12: The flow as a fraction of the asymptotic flow for $p = 1 + n = 1.2, 1.3, \ldots, 2.0$, where n is the power-law index.

n	A	B	C	D
0.2	0.1881	0.5323	0.7584	0.8998
0.3	0.2449	0.6327	0.8407	0.9468
0.4	0.2890	0.7035	0.8915	0.9711

Table 1: Average flow $\overline{v}$ as a fraction of the asymptotic flow in the regions A, B, C, and D for stationary flow in a rectangular channel for some values of the power-law index n.

6.2.2. Narrow channels

We let the weight function in a region with a narrow channel of width hl be $\frac{1}{h\overline{v}}$ where h is the thickness of the mould in the region and

$$\overline{v} = \frac{\int_0^l \int_0^1 u(x,z)\,dzdx}{l\int_0^1 u_a(z)\,dz}$$

is the average flow as a fraction of the asymptotic flow. The results in the table were computed using PLTMG and MATLAB by the author using program code written by Klas Samuelsson. Note that the average flow for a channel of width l is smaller than the reduced flow at a distance $\frac{1}{4}l$ from the "wall", as can be seen by comparing Figure 12 and Figure 13 (or Table 2). The comparison makes sense because, in a channel of width l, the average distance to the wall is clearly $\frac{1}{4}l$.

l	$\overline{v}$
0.5	0.017
1.0	0.11
2.0	0.35
3.0	0.51
4.0	0.62
6.0	0.74
8.0	0.80
16.0	0.89

Table 2: Average flow $\overline{v}$ as a fraction of the asymptotic flow for stationary flow in a rectangular channel with height 1.0 and width l for power-law index $n = 0.3$.

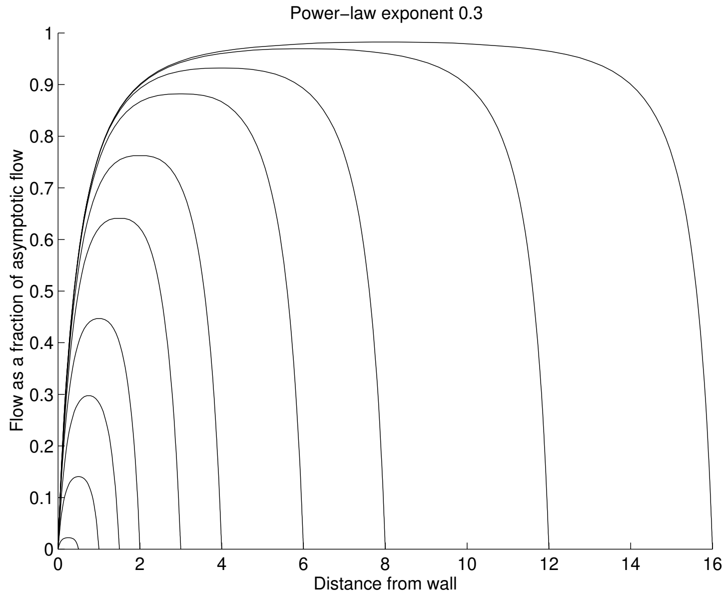


Figure 13: The flow as a fraction of the asymptotic flow for power-law index $n = 0.3$ and channel width $l = 0.5, 1, 1.5, 2, 3, 4, 6, 8, 12, 16$. Note the plug flow character for the bigger values of the width.

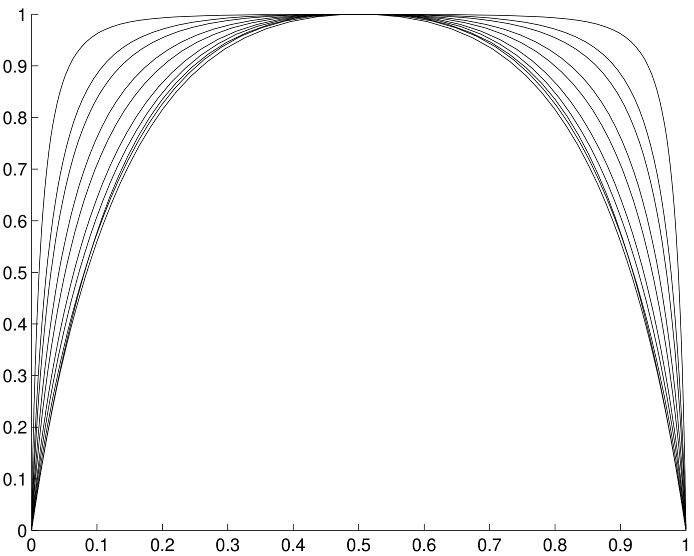


Figure 14: Normalizing the plot in Figure 13 by rescaling the x-coordinate to the interval $[0, 1]$ and the maximal flow to 1. The curve with most attenuated plug flow character is with width $l = 32$ and the curve with least plug flow character is with width $l = 0.5$.

7. Results of computations

7.1. Examples from Tadmor & Gogos

The geometry for the examples is described in Figures 15–17. The result of our algorithm for the distance function is shown in Figures 18–20. Constant inflow is assumed. The error due to discretization is probably very small, since reducing the discretization to values less than 0.04 for examples 1 and 2 and less than 0.2 for example 3 gives plots with no visible difference to the human eye.

In Figure 18 we can see that the distance model predicts the positions of flow-fronts with acceptable precision. Note that the flow-fronts for the short shots already at the beginning show a tendency to advance further at the upper side than the lower, which indicates some sort of asymmetry in the mould or at the injection gate. This could be part of the explanation for the difference in position of the weld-line predicted by the distance model and the actual weld-line. One other part is the over-estimation of the flow through the narrow passage under the obstacle that the distance model apparently gives. The FEM-computations were done on a relatively coarse grid and the scattered distribution of points on the flow-fronts gives a rough idea of the size of the error.

The second example, shown in Figure 19, has acceptable correspondence between the distance model and the first short shot where the flow-front in the narrower passage under the obstacle is slightly behind the flow-front above. But beyond the obstacle the distance model predicts less flow through the narrow passage than seems to be the case in the short shots; the weld line is predicted lower than actual position. This has not been observed in any other case and is so far unexplained. In the last short shot we can see that the lower part of the flow-front has advanced further than the upper. The FEM-computations seem to be of questionable accuracy, e.g. the first flow-front plotted does not even have circular shape.

In Figure 20 we can see that the distance model predicts weld lines from all corners inside the mould of the thinner regions, in fact we have backflow into those regions. This is not visible in the short shots or the FEM-simulations.

Figures 21–23 illustrate the effect of the discretization error. Note that in those figures level curves at equidistant weighted distance are plotted, i.e. not time scale.

The computations have been done on two PC:s, one with a 100 MHz Pentium CPU (PC1), the other with a 66 MHz 486DX2 CPU (PC2), both having 16 MB RAM, a Sun Sparc IPX workstation (IPX), also having 16 MB RAM and a Sun Sparcstation-5 workstation (SS-5) having 32 MB RAM.

For T&G, Ex. 1 with 21 polygons, 62 edges, and 42 vertices in the subdivision and $\Delta = 0.01$ the execution times are given in minutes:seconds and summarized in Table 3.

For T&G, Ex. 2 with 10 polygons, 31 edges, and 22 vertices in the subdivision and $\Delta = 0.01$ the execution times are summarized in Table 4.

For T&G, Ex. 3 with 7 polygons, 22 edges, and 16 vertices in the subdivision

h	PC1	PC2	SS-5	IPX	nodes
0.01	3:23	9:37	—	—	3283
0.02	1:43	4:28	8:21	14:55	1673
0.04	0:54	2:21	5:36	7:08	859
0.08	0:31	1:20	4:21	4:07	457
0.16	0:20	0:51	3:42	2:28	259
0.32	0:15	0:37	3:23	1:38	153

Table 3: Execution times for T&G, Ex. 1.

h	PC1	PC2	SS-5	IPX	nodes
0.005	6:02	11:36	—	—	2132
0.01	3:20	8:43	14:23	28:40	1082
0.02	1:44	4:32	8:35	14:35	557
0.04	0:55	2:22	5:46	7:48	291
0.08	0:32	1:20	4:28	4:25	160
0.16	0:21	0:51	3:44	2:48	95
0.32	0:16	0:36	3:26	1:39	61

Table 4: Execution times for T&G, Ex. 2.

and $\Delta = 0.04$ the execution times are summarized in Table 5. We note that

h	PC1	PC2	SS-5	IPX	nodes
0.05	4:31	12:18	—	40:55	1623
0.1	2:15	6:05	10:03	20:26	823
0.2	1:08	3:12	6:12	10:40	423
0.4	0:37	1:40	4:14	5:46	223
0.8	0:22	0:59	3:18	3:19	123
1.6	0:15	0:38	2:49	1:57	71

Table 5: Execution times for T&G, Ex. 3.

the PC:s are faster than the workstations. This can be an effect of the load on the network that the workstations are connected to and the fact that the PC:s have a local disc for storing data files while the workstations only have a local disc for swapping. Also the operating system on the PC:s, Linux, seems to be more efficient than the operating system on the workstations, Solaris. In any case, our algorithm seems to be well adapted to running fast on a PC.

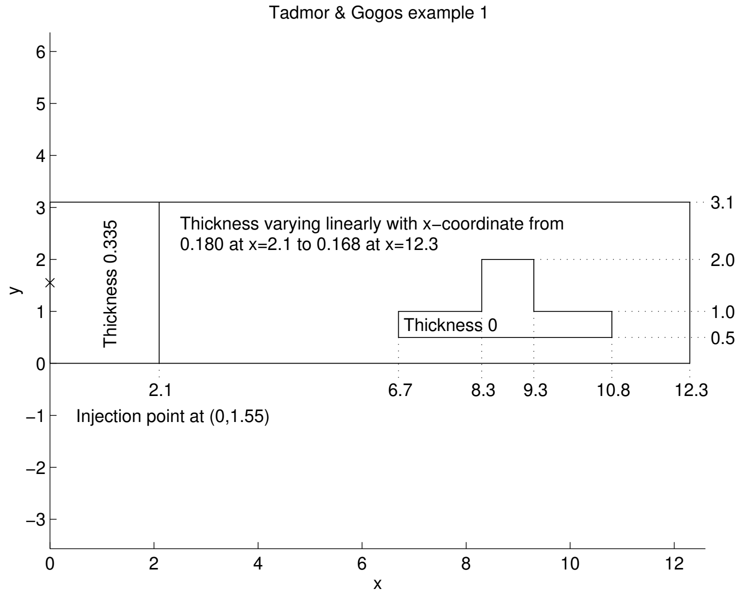


Figure 15: The geometry for T&G, Ex. 1.

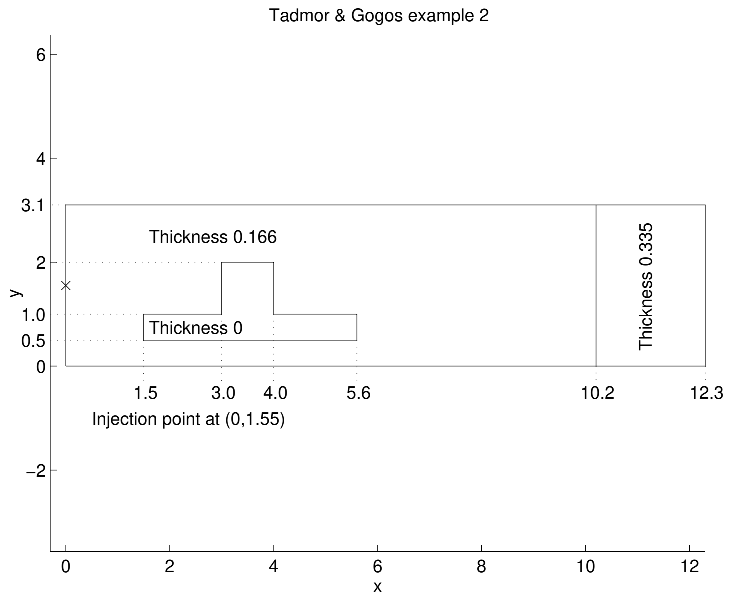


Figure 16: The geometry for T&G, Ex. 2.

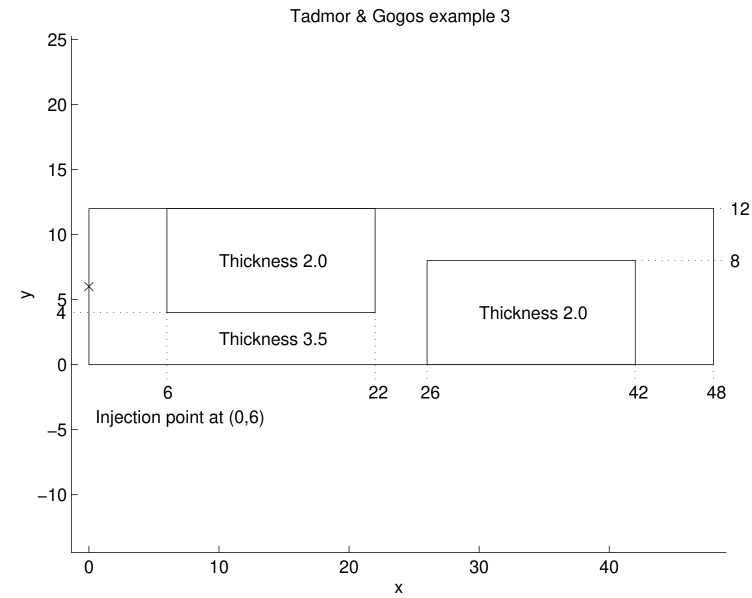


Figure 17: The geometry for T&G, Ex. 3.

1.
2.

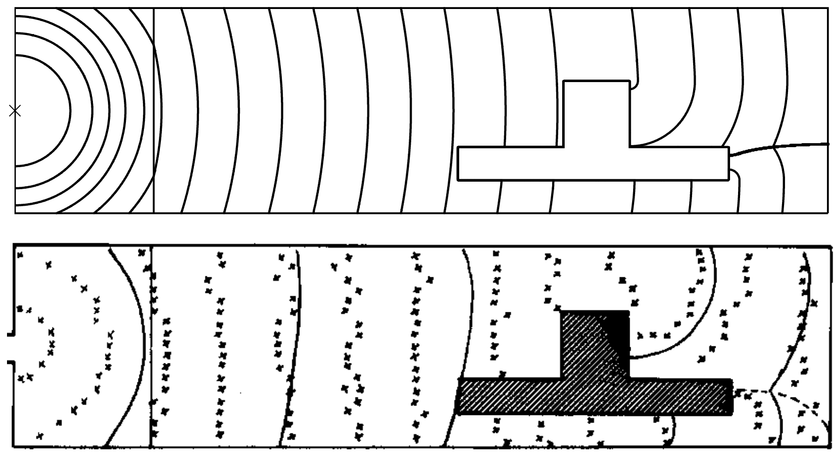
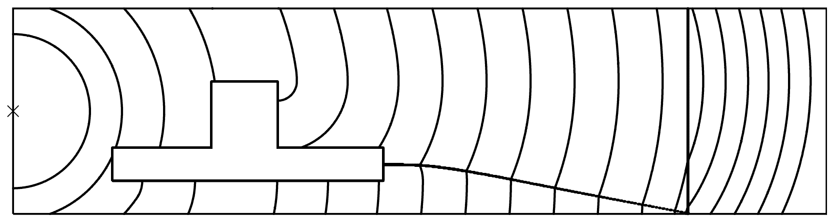


Figure 18: The flow-fronts and weld lines for T&G, Ex. 1 according to (1) the distance model with flow-fronts at equidistant time steps, (2) [49] where solid lines represent short shots and x stands for numerical simulations of isothermal non-Newtonian flow with a lattice-type FEM formulation on a computational grid of 18x72. The geometry is described in Figure 15.

1. 

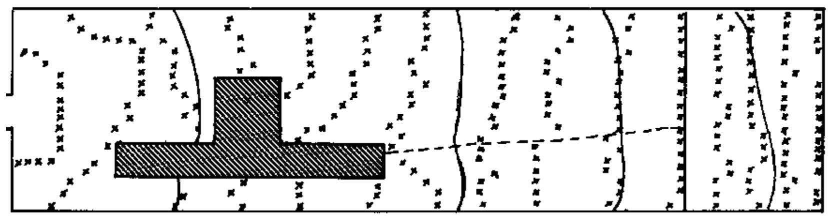
2. 

Figure 19: The flow-fronts and weld lines for T&G, Ex. 2 according to (1) the distance model, (2) [49] where solid lines represent short shots and x stands for numerical simulations of isothermal non-Newtonian flow with a lattice-type FEM formulation on a computational grid of 18x72. The geometry is described in Figure 16.

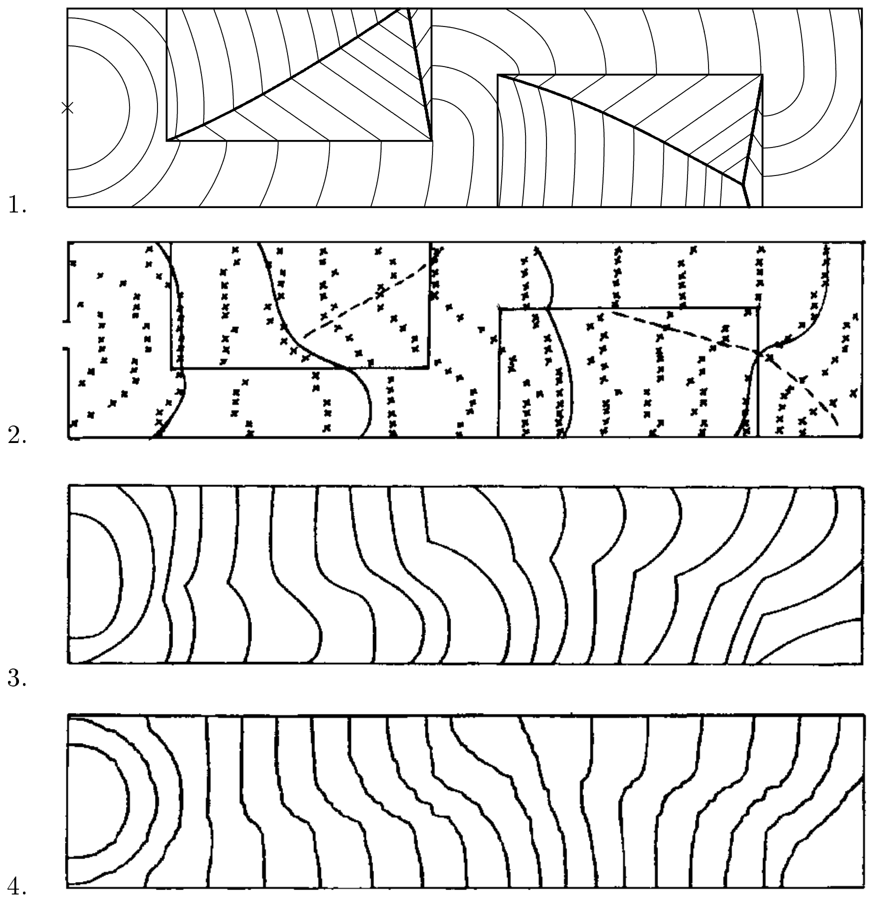
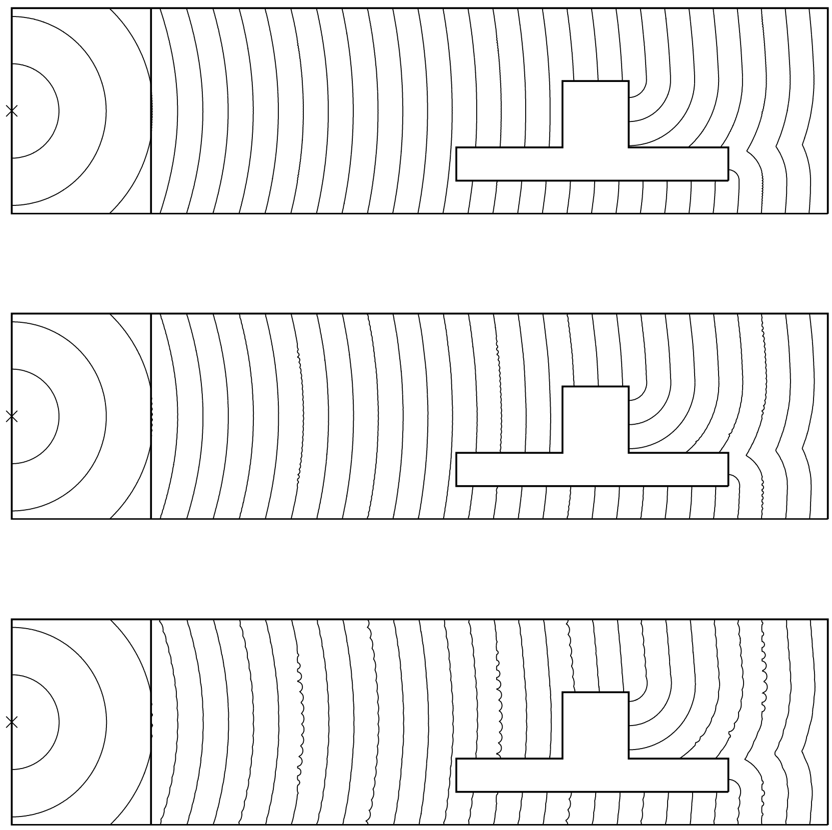


Figure 20: The flow-fronts and weld lines for T&G, Ex. 3 according to (1) the distance model, (2) experiments and isothermal simulations of non-Newtonian flow with a lattice-type FEM formulation on a computational grid of 18x72 in [49], (3) experiments with a Newtonian fluid in [31], (4) numerical simulation of isothermal Newtonian flow on a computational grid of 12x48 in [48]. The geometry is described in Figure 17.



1.

2.

3.

Figure 21: Level curves for the approximating distance function for T&G, Ex. 1 with discretization (1) 0.04, (2) 0.08, and (3) 0.16. The linearly varying thickness has been approximated by piecewise constant thickness in 11 regions. The discretization error is noticeable near the separation lines between these regions in cases 2 and 3. The geometry is described in Figure 15.

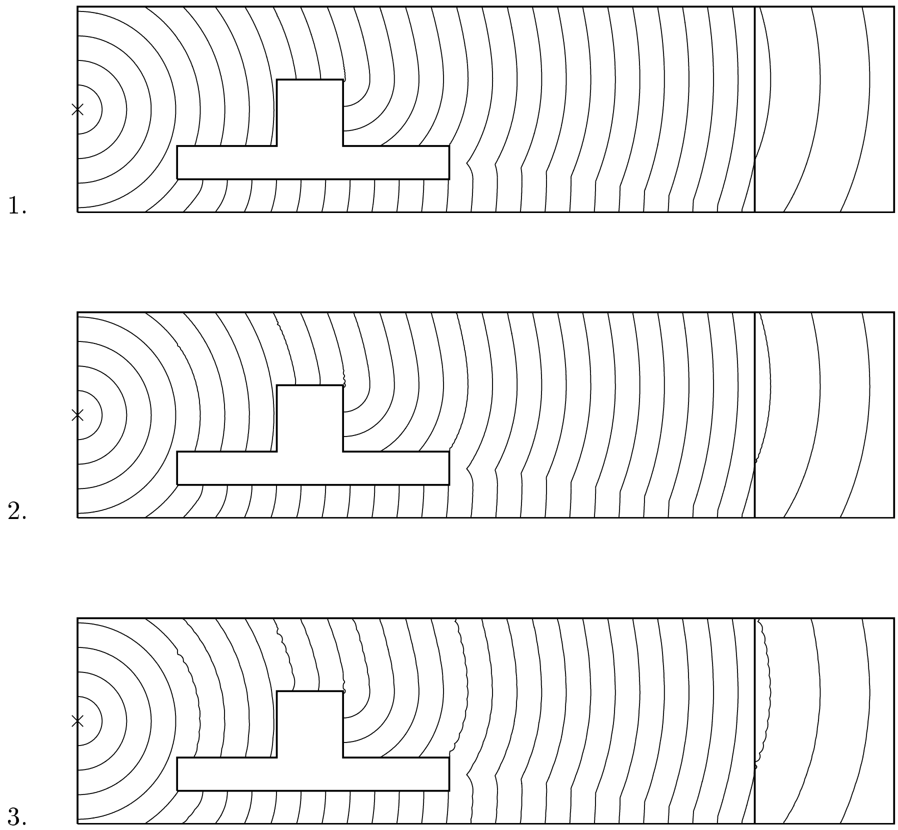


Figure 22: Level curves for the approximating distance function for T&G, Ex. 2 with discretization (1) 0.04, (2) 0.08, and (3) 0.16. Largest discretization errors occur near the separation lines. The effect can be described as introducing a grating at separation lines where the flow emanates as point sources. If the discretization is too coarse these point sources are separated enough to give a visible effect. The geometry is described in Figure 16.

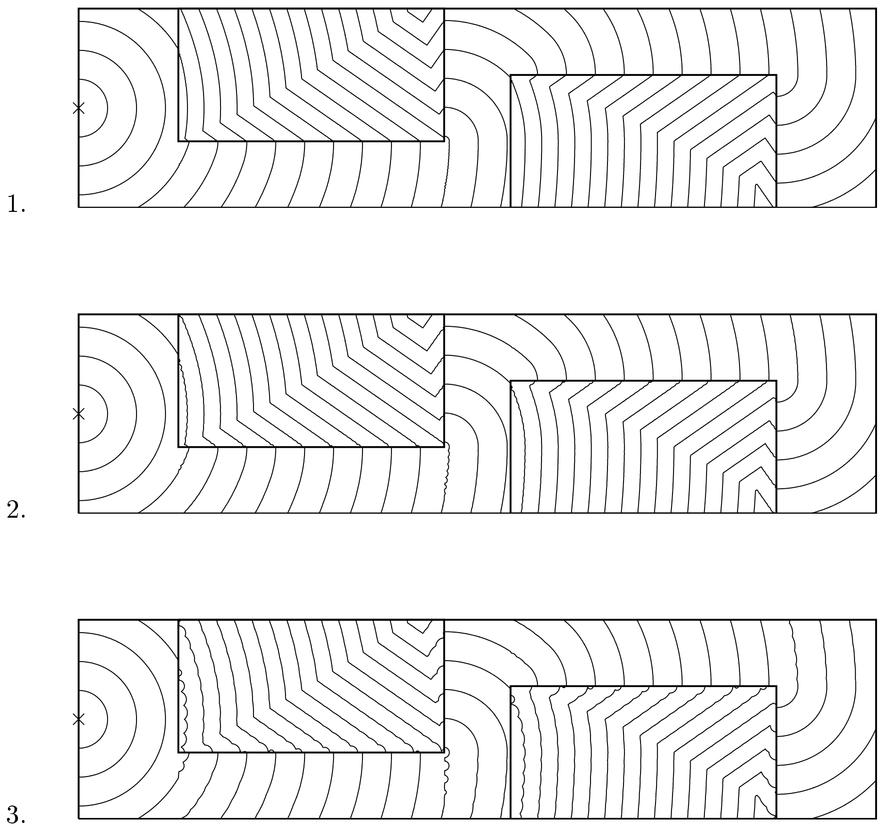


Figure 23: Level curves for the approximating distance function for T&G Ex. 3 with discretization (1) 0.2, (2) 0.4 and (3) 0.8. Largest discretization errors occur near separation lines. The geometry is described in Figure 16.

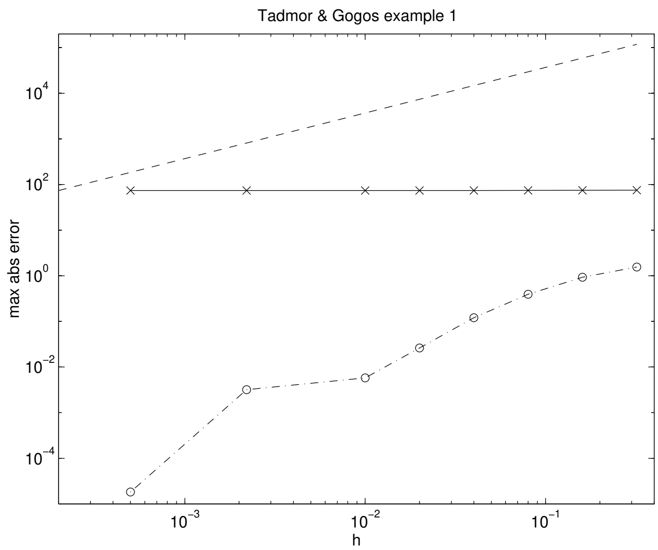


Figure 24: Discretization error as a function of discretization parameter h for T&G, Ex. 1. Note that the solid line, the sum of error estimate for finest discretization and the difference between the finest and current discretization, is dominated by the pessimistic estimate.

7.1.1. Analysis of discretization error

In Figures 24–26 dashed line is direct application of the theoretical bound (5.15). Dash-dotted line shows difference from computation with finest discretization, i.e. $\max_x |d_h(x) - d_{h_{min}}(x)|$. Solid line shows the sum of the theoretical bound for finest discretization and the difference between the finest discretization and the current discretization, which is a true error estimate since $d_h(x) - d(x) \leq |d_h(x) - d_{h_{min}}(x)| + |d_{h_{min}}(x) - d(x)|$. For T&G, Ex.1 we do not get a useful error estimate since our theoretical error estimate is too pessimistic. In Figures 25 and 26 we can see that the theoretical error estimate is pessimistic in those cases.

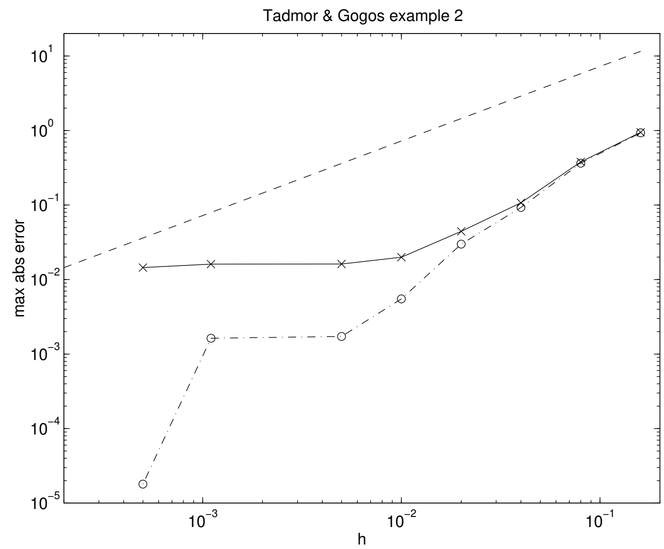


Figure 25: Discretization error as a function of discretization parameter h for T&G, Ex. 2.

7.2. Example from Menges et al [35]

The result from the distance model with constant inflow is shown in Figure 27. Comparison can be made with experimental results in figure 2 in [35], p. 672. One can see that in the experimental result the flow-fronts in the wider part are closer to fulfilling the boundary condition of orthogonality to the boundary than in the distance model.

7.3. Example from Crochet et al [1], pp. 456–459

The result from the distance model with constant inflow is shown in Figure 28. The geometry is a simple model of a dashboard. Here we have two injection

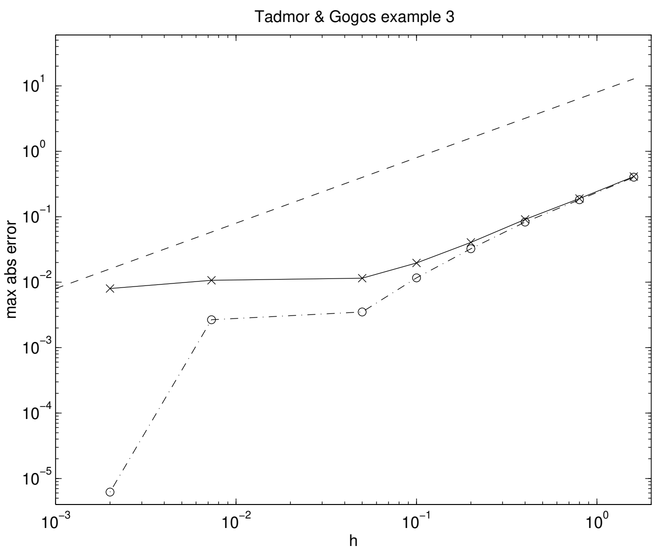


Figure 26: Discretization error as a function of discretization parameter h for T&G, Ex. 3.

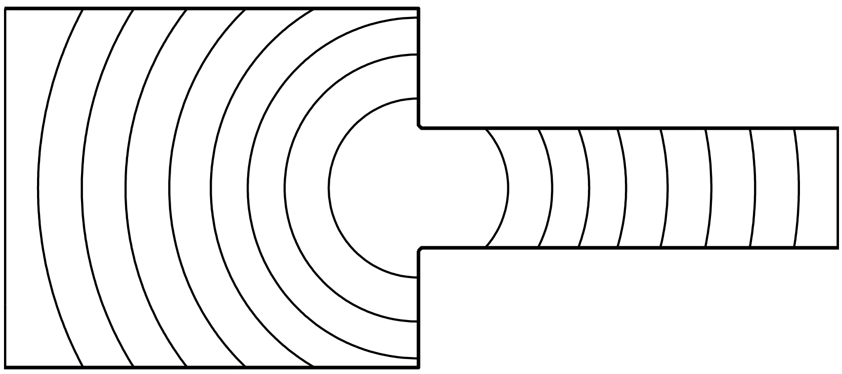


Figure 27: The flow-fronts according to the distance model with flow-fronts at equidistant time steps. Example from Menges et al [35].

points and this is a purely numerical experiment where we get the flow-fronts by computing the shortest distance to any of the two injection points. Here we have weld lines occuring both due to inserts in the mould and due to the double injection points.

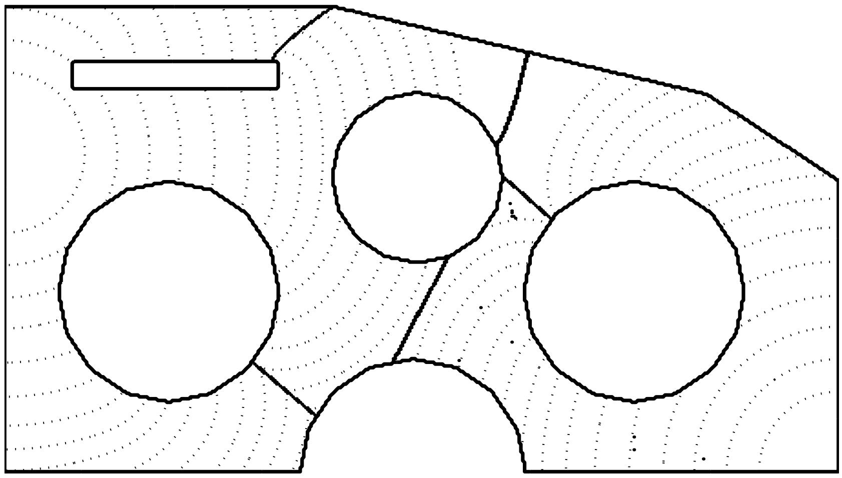


Figure 28: The flow-fronts and weld lines according to the distance model with flow-fronts at equidistant time steps. Example from Crochet et al, [1], pp. 456–459.

7.4. Example from Boshouwers and van der Werf [10]

The geometry is described in Figure 29. The result from the distance model with constant inflow is shown in Figure 30. Comparison can be made with figures in [10], pp. 121–124.

For more examples where the distance model successfully predicts the filling pattern see [9].

7.5. Computations including boundary influence

Figures 31–37 show the results from the distance model with and without compensation for boundary influence as described in Section 6.2.1 with $n = 0.3$ using the values from Table 1 in that section.

We can see that in Figures 31 and 32 the weld line does not move noticeable when taking the boundary effects into account. But naturally the flow-fronts do get affected, they are clearly retarded near the boundary. It is not so surprising that the weld line is not affected if one observes that in this geometry two shortest paths to a point on the weld line stay near the boundary of the obstacle on equally long parts. By "forcing" the shortest paths away from the obstacle the paths get roughly equally prolonged.

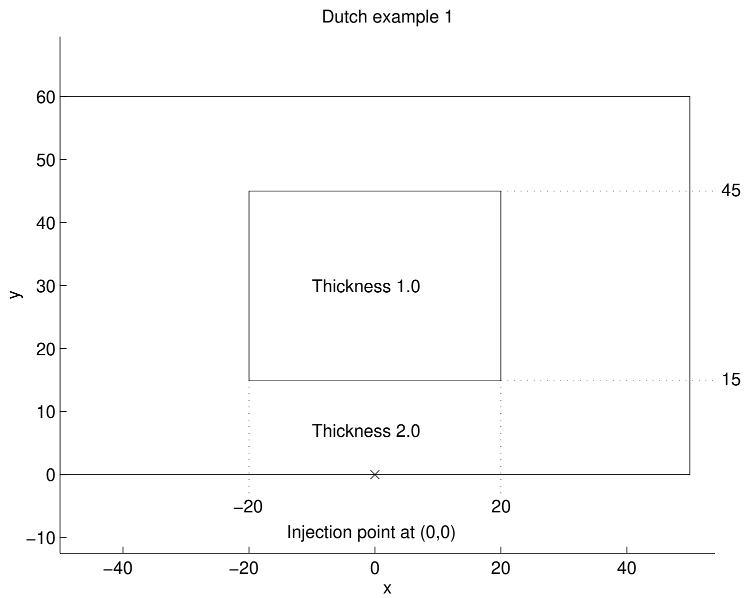


Figure 29: The geometry.

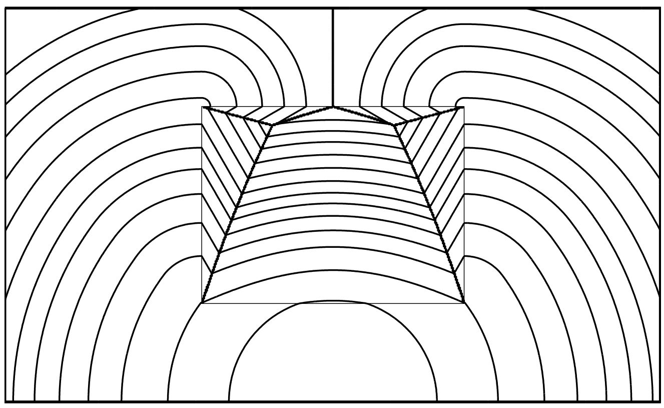
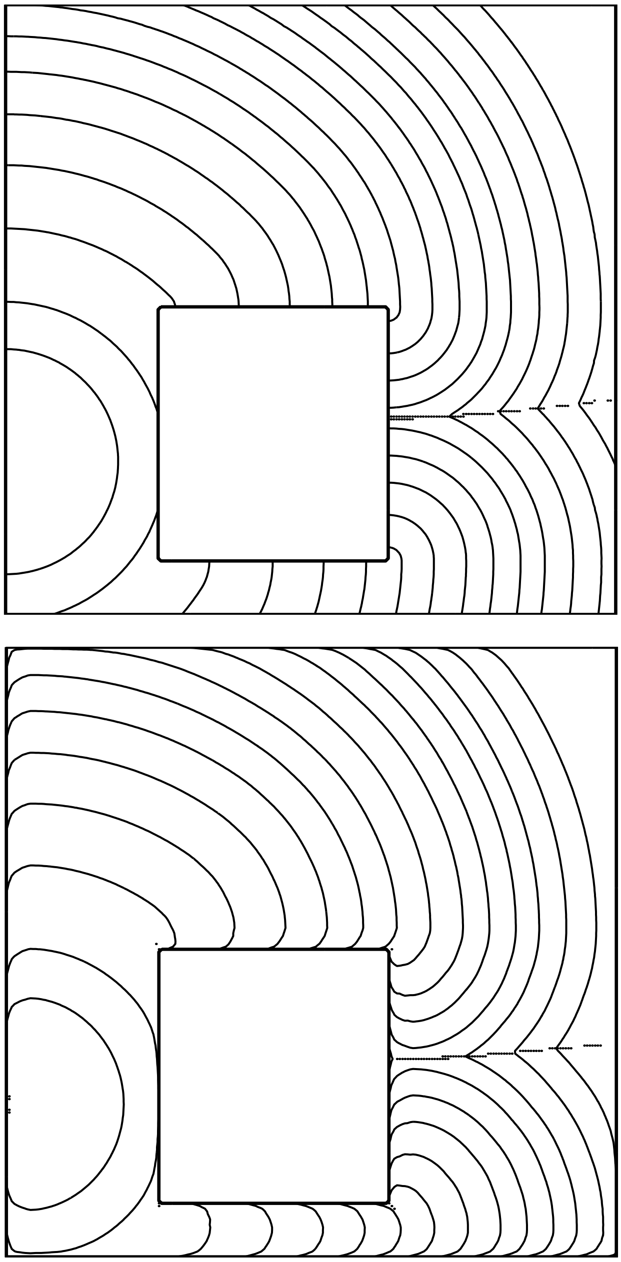


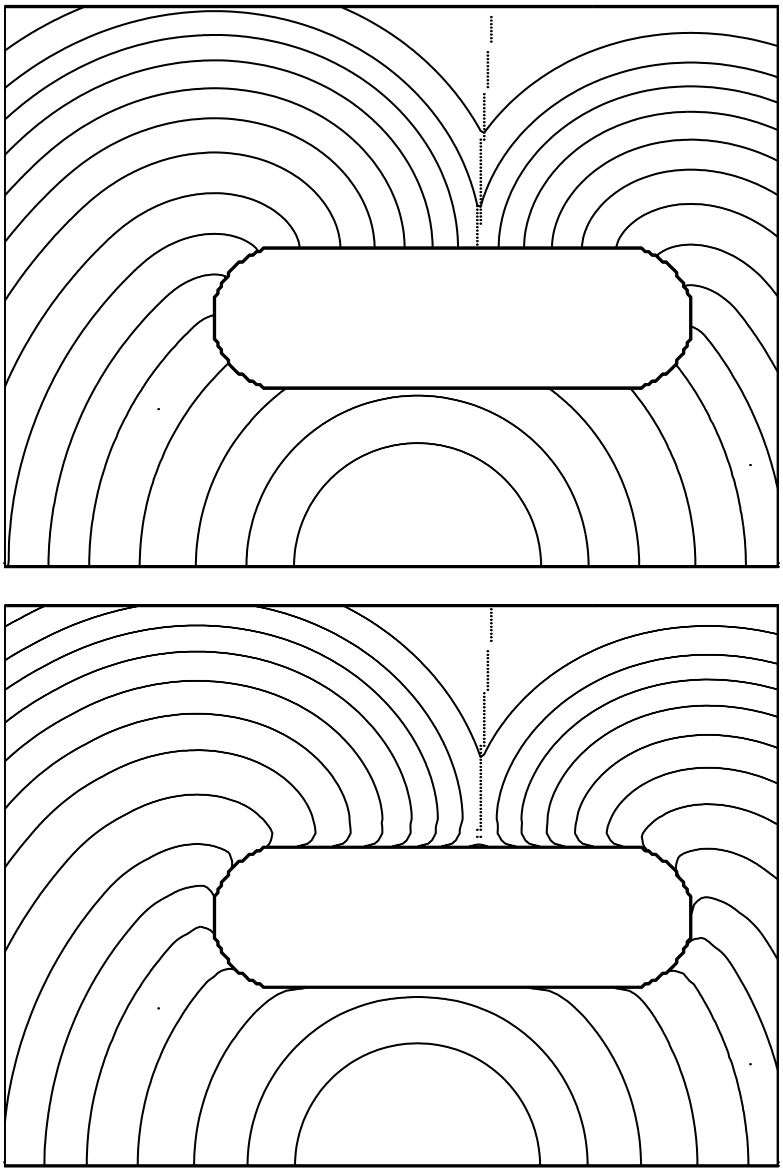
Figure 30: The flow-fronts and weld lines according to the distance model with flow-fronts at equidistant time steps.



1.

2.

Figure 31: Results from distance model (1) without boundary influence and (2) with boundary influence.



1.

2.

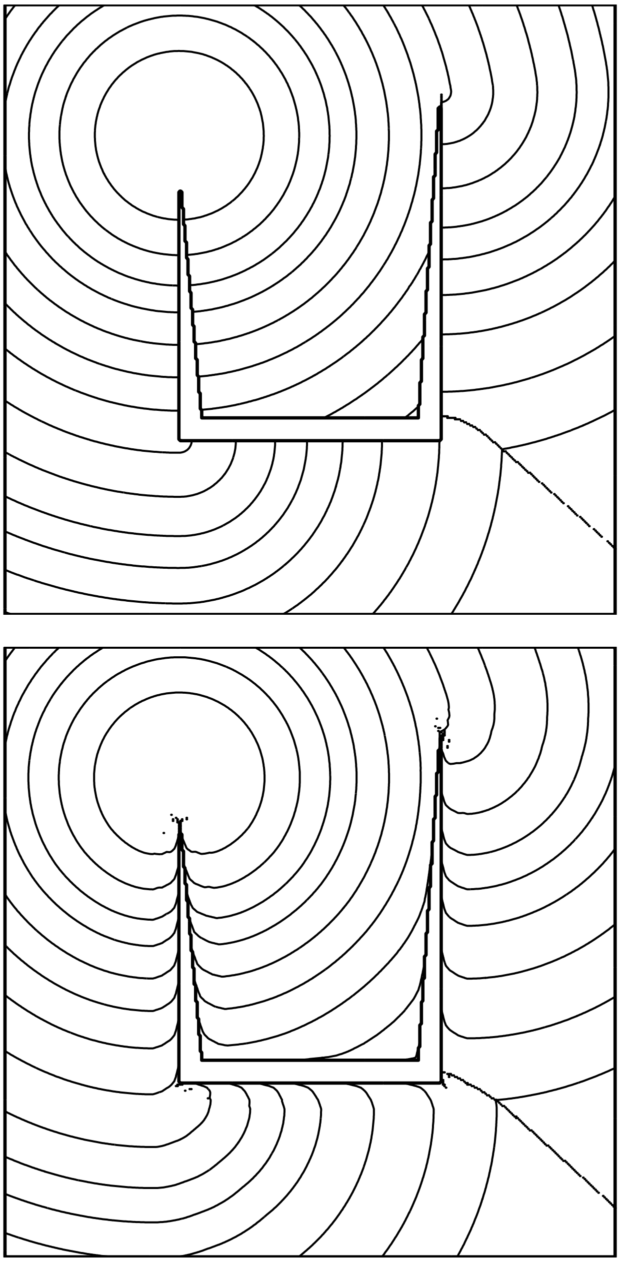
Figure 32: Results from distance model (1) without boundary influence and (2) with boundary influence only at the inner boundary.

In Figure 33 one can see that the weld line intersects the obstacle a little bit closer to the lower rightmost corner when taking the boundary effects into account. The movement of the weld line is however only about 2% of the relevant Euclidean distance. This example is constructed so that a significantly larger part of one of the paths to a point on the weld line runs near the boundary of the obstacle.

When the weight is high near the boundary the shortest path is "forced" to go a bit away from the boundary. This results in a prolongation that is less than the increase in weighted length that would occur for the path that was shortest in the non-weighted case. This prolongation is at each corner proportional to the angle minus π, see Figure 34. In Figure 35 we can see that the weld line intersects the obstacle at a point closer to the lower rightmost corner of the triangle when we take the boundary effect into account. The movement of the weld line is about 4% of the actual Euclidean distance. The effect is similar in Figure 36 where the weld line moves about 3% towards the upper rightmost corner of the obstacle.

We can now construct a geometry where the effect is more significant; see Figure 37 where the weld line moves about 6% of the interior Euclidean distance from the injection point to the weld line.

The layers near the boundary also give rise to false weld lines due to the discontinuity in the weight across the boundary of the layers. This gives weld lines of the same type as in the thinner region in Figures 30 and 20. In the case of taking boundary effects into account these weld lines are not so pronounced and sometimes only a few dots are detected. The effect is visible in Figures 31 and 33 as a few dots near the corners of the obstacle, in Figures 35 and 36 also near the injection point and finally in Figure 37 there is quite a lot of false weld line detection, also along the edges of the boundary. The effect depends much on the geometry and the position of the injection point.



1.

2.

Figure 33: Results from distance model (1) without boundary influence and (2) with boundary influence.

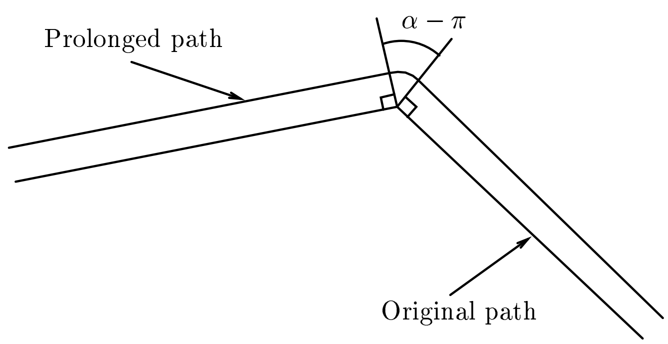
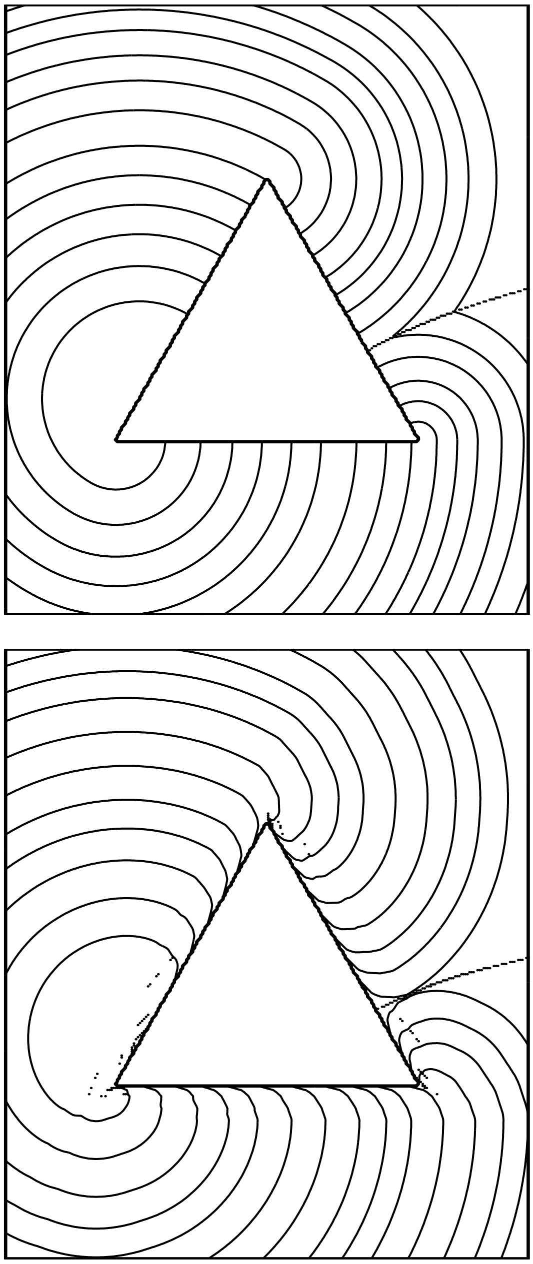


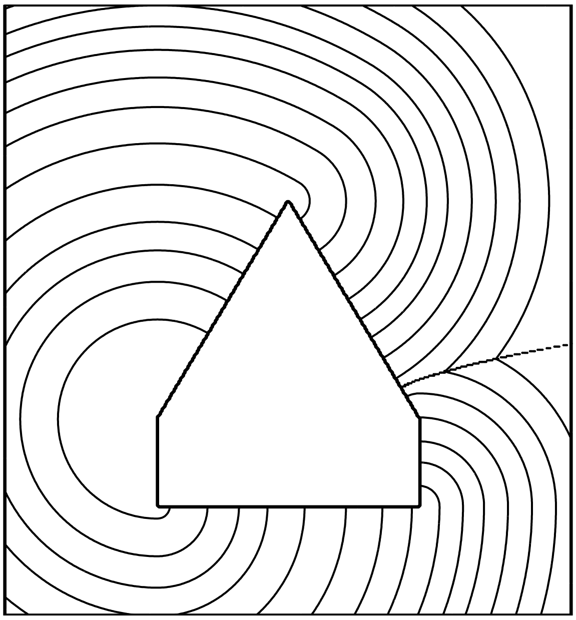
Figure 34: The path is prolonged an amount proportional to $\alpha - \pi$



1.

2.

Figure 35: Results from distance model (1) without boundary influence and (2) with boundary influence.

1. 

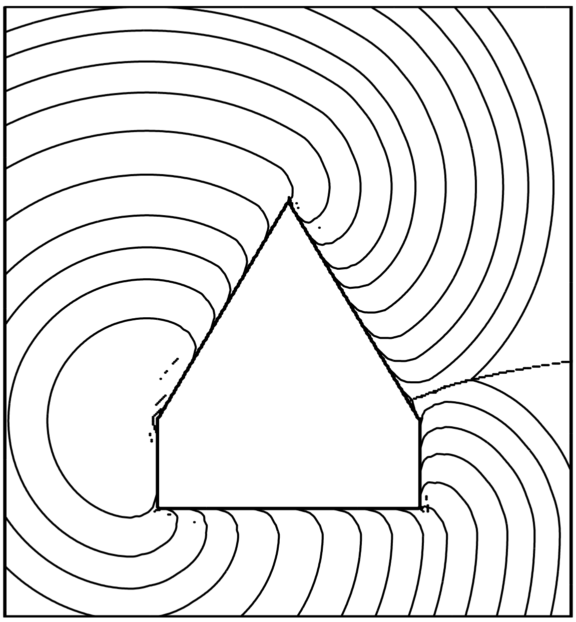
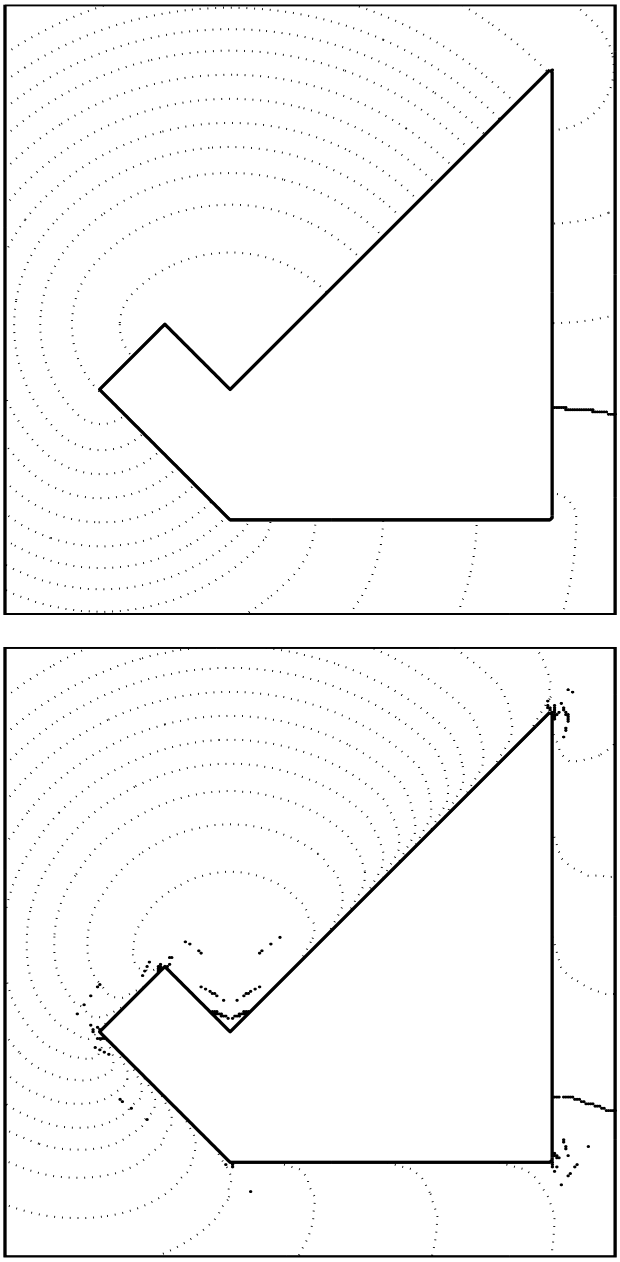
2. 

Figure 36: Results from distance model (1) without boundary influence and (2) with boundary influence.

1.



2.

Figure 37: Results from distance model (1) without boundary influence and (2) with boundary influence.

8. Principles for visualization

8.1. Adjusting time scale when inflow is constant

We will consider here the relation between distance and time in our distance model. We assume that the inflow q is constant, i.e. the volume of polymer melt injected into the mould per second is constant. Let $h : \Omega \to R$ be the mould thickness function. Clearly, the function

$$f(C) = \frac{1}{q} \int_{\{x \in \Omega : d(x) \leq C\}} h(x)\, dx$$

describes the time it takes for the flow-front to reach the position

$$\{x \in \Omega : d(x) = C\}.$$

By finding the inverse function $f^{-1}(t)$ we can obtain suitable distance values for plotting level curves corresponding to given time values. A procedure for finding the inverse function for equidistant time values was implemented by Lundquist in [33] as a MATLAB-function called `filltime`. He used the secant method to find a solution C to the equation $f(C) = t$.

8.2. Finding weld lines by the arrival angle method

In this section we will describe a method for visualizing weld lines. Recalling Definition 11 of arrival angle in Section 4.6.2, we realize that if d is differentiable at x then the arrival angle at x is $\arg \nabla d(x)$. Thus to detect points on a weld line we will search for points where the arrival angle has a discontinuity. According to Proposition 4.10 this method will also detect points where the weight function has discontinuities, i.e. where we have abrupt changes in the mould thickness, so-called junctions. Since we only calculate the values of the distance function on a discrete grid we detect the discontinuity by setting a threshold for the difference in arrival angle for neighbouring points. By marking all points that have numerical gradient[6] with components exceeding this threshold[7] value we also get some extra points marked where the arrival angle varies fast, e.g. at the injection point or at flow around a corner. The discretization of separation lines can also give rise to large variations in the arrival angle for points near the separation line.

Another method for visualizing the discontinuities is to map the arrival angle to a colour in a cyclic spectrum obtained from linear interpolation in RGB-space between a few colours, e.g. red, green and blue. This has the disadvantage that a jump between two different colours is not always clearly visible. It seems to be difficult to choose a set of colours such that the mapping from arrival angle to the colours always gives rise to a visible contrast for a fixed angle difference. This means that maybe not all weld lines will be clearly visible.

[6] Using the MATLAB command gradient.

[7] At least one component with magnitude greater than the threshold.

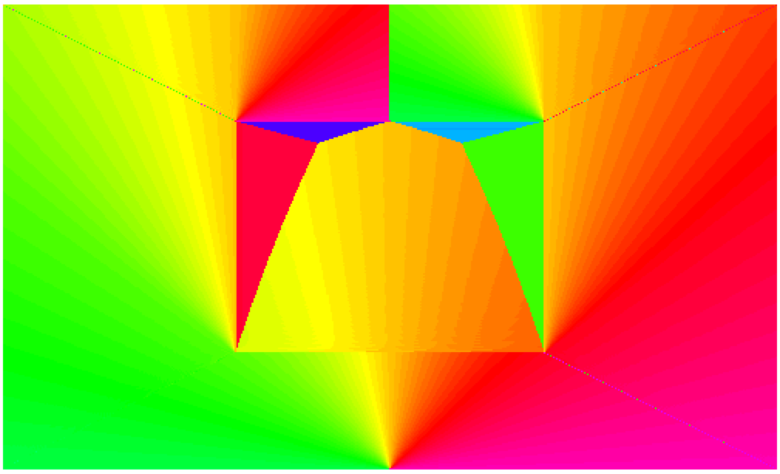


Figure 38: The arrival angle visualized with colour for the example from Boshouwers and van der Werf, see Section 7.4 and Figure 30.

However, by letting the mapping depend on time so that the colour scale is rotating cyclically each weld line (or part of it) will be very clearly visible at certain moments. This visualization requires some explanation but can be very suggestive. An advantage is that there is no need for determining a threshold value and the “false” detection of weld lines is not disturbing.

9. Conclusions

9.1. Discussion

The algorithm described in this thesis has been implemented as a computer program, compiled and tested on Sun workstations with the operating system Solaris, DEC workstations with the operating system OSF1, PC of IBM-compatible type with the free operating system Linux as well as DOS/Windows. The implementation is written with portability in mind. The program uses a simple input file format that is tedius to construct "by hand", therefore Fredrik Berntsson wrote a graphical user interface using MATLAB. The interface uses mouse or keyboard for input of the geometry and saves the input data file for the distance function calculation as well as allows the user to start the program from a menu option. Further, menu options allow the user to call routines for visualization of the results. Thus we have a complete system for entering geometries, computing the distance function and visualizing the results. The system can be improved by e.g.

1. including more functions for editing an existing geometry,
2. visualizing non-planar geometries in 3D, and not only as a lay-flat model, and
3. adding functions to import geometry data from some existing CAD system.

These improvements are however not of academic research interest, but essential if a commercial product is to be developed or if the system is to be of practical use for industry.

The algorithm runs fast on the relatively simple geometries we have tried so far and the results from the computations seem very reasonable. Our system can of course not compete with the advanced commercial systems available from e.g. Moldflow, I-DEAS, and C-MOLD, since they offer a system for all stages in the design and not only, as we do, analyze the filling stage. However, some critical parameters can be investigated in a preliminary study by using our program. Further, the designers can, by using our program, gain a rough but very quick understanding of the problems in the mould design phase and thus hopefully avoid some later problems.

9.2. Recommendations for further study

Further investigations regarding execution time on more complex geometries and the performance of alternative algorithms for solving the WRP can be done. Methods for increasing the computational speed by improving our algorithm are also available, currently we have not found this to be urgent since the program is fast enough on the geometries we have investigated so far.

Thermal effects like viscous heating and cooling from the mould can probably be incorporated in our method, but at the cost of a more complex algorithm.

The question is then: how severe complications are necessary for having a meaningful thermal analysis? As usual, it will be a trade-off between simplicity and accuracy.

More systematic comparisons with results from *both* experiments and simulations using commercial or research packages are also recommended. Besides the well known commercial packages for possible comparisons, there are research programs written at several European universities. One example is Hele–Shaw Fill within the Diffpack project at SINTEF, Oslo, Norway.

It seems that only the biggest research centres have resources for really systematic investigations here. We have not had such big resources.

The most important improvement would anyhow be to develop a thermal calculation that does not destroy the simplicity of the distance model.

References

1. Advani, S. G. *Flow and Rheology in Polymer Composites Manufacturing.* Elsevier, Amsterdam, 1994.

2. Aigner, M. Implementering och testning av en modell för formsprutning av plast. Master's thesis, LiTH-MAT-EX-92-01, Linköping, 1992. (In Swedish).

3. Andersson, T. FEM-solution of the p-Poisson equation in 2D. Master's thesis, LiTH-MAT-EX-90-03, Linköping, 1990.

4. Aronsson, G. Stationary pipe flow of power-law fluids. In *Proceedings of the Fifth European Conference on Mathematics in Industry*, M. Heiliö, Ed. Teubner, Stuttgart, and, Kluwer, Dordrecht, 1991, pp. 135–138.

5. Aronsson, G. On some properties of a distance function and a connection to injection molding. Preprint, Linköping, 1995.

6. Aronsson, G. On p-harmonic functions, convex duality and an asymptotic formula for injection mould filling. *European J. Appl. Math. 7* (1996), 417–437.

7. Aronsson, G., and Janfalk, U. On Hele–Shaw flow of power-law fluids. *European J. Appl. Math.* (1992), 343–366.

8. Asano, T., Asano, T., Guibas, L., Hershberger, J., and Imai, H. Visibility of disjoint polygons. *Algorithmica 1* (1986), 49–63.

9. Berntsson, F. Simulation tools for injection moulding. Master's thesis, LiTH-MAT-EX-96-12, Linköping, 1996.

10. Boshouwers, A. H. M., and van der Werf, J. J. *Inject-3, a simulation code for the filling stage of the injection moulding process of thermoplastics.* PhD thesis, Eindhoven, 1988.

11. Busemann, H. *The Geometry of Geodesics.* Academic Press, New York, 1955.

12. Cherkassky, B. V., Goldberg, A. V., and Radzik, T. Shortest paths algorithms: Theory and experimental evaluation. In *Proceedings of the Annual ACM SIAM Symposium on Discrete Algorithms.* Assoc. Comput. Mach., New York, 1994, pp. 516–525.

13. Dijkstra, E. W. A note on two problems in connexion with graphs. *Numer. Math. 1* (1959), 269–271.

14. Dreyfus, S. E. An appraisal of some shortest-path algorithms. *Oper. Res. 17* (1969), 395–412.

15. Edelsbrunner, H., and Guibas, L. Topologically sweeping an arrangement. *J. Comput. System Sci. 38* (1989), 165–194.

16. Edelsbrunner, H., and Guibas, L. Corrigendum: "Topologically sweeping an arrangement". *J. Comput. System Sci. 42* (1991), 249–251.

17. Edelsbrunner, H., O'Rourke, J., and Seidel, R. Constructing arrangements of lines and hyperplanes with applications. *SIAM J. Comput. 15* (1986), 341–363.

18. Ekeland, I., and Temam, R. *Convex Analysis and Variational Problems.* North-Holland, Amsterdam, and, Elsevier, New York, 1976.

19. Evans, J. R., and Minieka, E. *Optimization Algorithms for Networks and Graphs.* Marcel Dekker, New York, 1992.

20. Fredman, M. L., and Tarjan, R. E. Fibonacci heaps and their uses in improved network optimization algorithms. *J. Assoc. Comput. Mach. 34* (1987), 596–615.

21. Ghosh, S. K., and Mount, D. M. An output-sensitive algorithm for computing visibility graphs. *SIAM J. Comput. 20* (1991), 888–910.

22. Gibbons, A. *Algorithmic Graph Theory.* Cambridge Univ. Press, London, 1985.

23. Hershberger, J., and Suri, S. Efficient computation of Euclidean shortest paths in the plane. In *Annual Symposium on Foundations of Computer Science.* IEEE, Computer Society Press, Los Alamitos, Calif., 1993, pp. 508–517.

24. Hieber, C. A., and Shen, S. F. A finite-element/finite-difference simulation of the injection-molding filling process. *J. Non-Newtonian Fluid Mech. 7* (1980), 1–32.

25. Hung, M. S., and Divoky, J. J. A computational study of efficient shortest path algorithms. *Comput. Oper. Res. 15* (1988), 567–576.

26. Janfalk, U. *On certain problems concerning the p-Laplace operator.* Ph.d. dissertation, Linköping, 1993.

27. Janfalk, U. Behaviour in the limit, as $p \to \infty$, of minimizers of functionals involving p-dirichlet integrals. *SIAM J. Math. Anal. 27* (1996), 341–360.

28. Kennedy, P. *Flow Analysis of Injection Molds.* Carl Hanser Verlag, München, 1995.

29. Kindl, M. R., Shing, M.-T., and Rowe, N. C. A stochastic approach to the weighted-region problem: I. the design of the path annealing algorithm. Tech. Report NPSCS-91-014, Dept. of Computer Science, Naval Postgraduate School, Monterey, Calif., 1991.

30. Knuth, D. E. *Sorting and Searching*, vol. 3 of *The Art of Computer Programming.* Addison-Wesley, Reading, Mass., 1973.

31. Krueger, W. L., and Tadmor, Z. Injection molding into a rectangular cavity with inserts. *Polym. Eng. Sci. 20* (1980), 426–431.

32. Lozano-Pérez, T., and Wesley, M. A. An algorithm for planning collision-free paths among polyhedral surfaces. *Comm. ACM 22* (1979), 560–570.

33. Lundquist, M. An implementation of a method for simulating injection moulding of non-planar objects. Master's thesis, LiTH-MAT-EX-95-09, Linköping, 1995.

34. Manzione, L. T. *Applications of Computer Aided Engineering in Injection Moulding.* Carl Hanser Verlag, München, 1987.

35. Menges, G., Lichius, U., and Bangert, H. Eine einfache Methode zur Vorausbestimmung des Fließfrontverlaufes beim Spritzgießen von Thermoplasten. *Plastverarbeiter 31* (1980), 671–676.

36. Mitchell, J. S. B. Algorithmic approaches to optimal route planning. *Proc. of SPIE - The Int. Society for Optical Engineering 1388* (1991), 248–259.

37. Mitchell, J. S. B. A new algorithm for shortest paths among obstacles in the plane. *Ann. of Math. and Artificial Intelligence 3* (1991), 83–106.

38. Mitchell, J. S. B. Shortest paths among obstacles in the plane. In *Proceedings of the 9th Annual Symposium on Computational Geometry.* ACM, New York, 1993, pp. 308–317.

39. Mitchell, J. S. B., and Keirsey, D. M. Planning strategic paths through variable terrain data. *Proc. SPIE Applications of Artificial Intelligence 485* (1984), 172–179.

40. Mitchell, J. S. B., Mount, D. M., and Papadimitriou, C. H. The discrete geodesic problem. *SIAM J. Comput. 16* (1987), 647–688.

41. Mitchell, J. S. B., and Papadimitriou, C. H. The weighted region problem: Finding shortest paths through a weighted planar subdivision. *J. Assoc. Comput. Mach. 38* (1991), 18–73.

42. Mitchell, J. S. B., Payton, D. W., and Keirsey, D. M. Planning and reasoning for autonomous vehicle control. *International Journal of Intelligent Systems II* (1987), 129–198.

43. Pettersson, L. FEM-lösning av p-Poissons ekvation i planet. Master's thesis, LiTH-MAT-EX-89-01, Linköping, 1989. (In Swedish).

44. Ragnemalm, I. *The Euclidean Distance Transform.* PhD thesis, Linköping Studies in Science and Technology, Dissertations No. 304, Linköping, 1993.

45. ROWE, N. C., AND RICHBOURG, R. F. An efficient Snell's law method for optimal-path planning across multiple two-dimensional, irregular, homogeneous-cost regions. *Internat. J. Robotics Res. 9* (1990), 48–66.

46. SAMUELSSON, K. *Numerical Solution of Pipe Flow Problems for Generalized Newtonian Fluids.* Licentiate thesis, Linköping Studies in Science and Technology, Thesis No. 367, LiU-TEK-LIC-1993:09, Linköping, 1993.

47. STORER, J. A., AND REIF, J. H. Shortest paths in the plane with polygonal obstacles. *J. Assoc. Comput. Mach. 41* (1994), 982–1012.

48. SYRJÄLÄ, S. *Numerical modelling of injection mould filling and cooling.* Licentiate thesis, Tampere, 1991.

49. TADMOR, Z., AND GOGOS, C. G. *Principles of Polymer Processing.* John Wiley & sons, New York, 1979.

50. TUCKER III, C. L. *Fundamentals of Computer Modeling for Polymer Processing.* Carl Hanser Verlag, München, 1989.

51. WELZL, E. Constructing the visibility graph for n line segments in $O(n^2)$ time. *Inform. Process. Lett. 20* (1985), 167–171.

52. ZIEMER, W. P. *Weakly Differentiable Functions: Sobolev Spaces and Functions of Bounded Variation.* Springer-Verlag, New York, 1989.