הטכניון
מכון טכנולוגי לישראל
TECHNION
Israel Institute of Technology

ספריות הטכניון
The Technion Libraries

בית הספר ללימודי מוסמכים ע"ש ארווין וג'ואן ג'ייקובס
Irwin and Joan Jacobs Graduate School

2D Simulation and Mapping using the Cauchy-Green Complex Barycentric Coordinates

Aviv Segall

2D Simulation and Mapping using the Cauchy-Green Complex Barycentric Coordinates

Research Thesis

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

Aviv Segall

Submitted to the Senate
of the Technion — Israel Institute of Technology
Tamuz 5776 Haifa July 2016

This research was carried out under the supervision of Prof. Mirela Ben-Chen, in the Faculty of Computer Science.

Some results in this thesis have been published as articles by the author and research collaborators in conferences and journals during the course of the author's master research period, the most up-to-date versions of which being:

Aviv Segall and Mirela Ben-Chen. Iterative Closest Conformal Maps between Planar Domains. *Computer Graphics Forum*, 2016.

Aviv Segall, Orestis Vantzos and Mirela Ben-Chen. Hele-Shaw Flow Simulation with Interactive Control using Complex Barycentric Coordinates. *Proceedings of the 15th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2016.

Acknowledgements

I would like to thank my advisor, my parents and my friends for all the support you gave me along the way.

I would also like to acknowledge ISF grant 699/12, Marie Curie CIG 303511, and the German-Israeli Foundation for Scientific Research and Development, Grant No: I-2378-407.6

The generous financial help of the Technion is gratefully acknowledged.

Contents

List of Figures

Abstract

Conformal maps are especially useful in geometry processing for computing shape preserving deformations, image warping and manipulating harmonic functions. The Cauchy-Green coordinates are complex-valued barycentric coordinates, which can be used to parameterize a space of conformal maps from a planar domain bounded by a simple polygon. In this work, we use the Cauchy-Green coordinates to simulate 2D potential flow with interactive control, and to construct conformal maps between planar domains.

The Hele-Shaw flow describes the slow flow of a viscous liquid between two parallel plates separated by a small gap. In some configurations such a flow generates instabilities known as Saffman-Taylor fingers, yielding intricate visual patterns which have been an inspiration for artists, yet are quite difficult to simulate efficiently. Formulating the equations with our framework allows us to efficiently simulate the flow and to provide the user with interactive control over the behavior of the fingers. Additionally, we show that the Cauchy-Green coordinates are applicable to the exterior of the domain, and use them for simulating two fluids with different viscosities.

The Riemann mapping theorem guarantees that there exists a conformal map between any two simply connected planar domains, yet computing this map efficiently is challenging. One of the main challenges is finding the boundary correspondence between the two domains. We use the Cauchy-Green coordinates for parameterizing the space of conformal maps from the source domain, and propose an alternating minimization algorithm for constructing a boundary-approximating conformal map, which implicitly finds a boundary correspondence. We enrich the space of solutions by generalizing the setup to quasi-conformal maps, and allow the user to interactively control the result using point-to-point and stroke-to-stroke constraints. Finally, we show applications to stroke based deformation and constrained texture mapping.

Chapter 1

Introduction

Many applications in computer graphics often involve computation which has to be carried out over a large domain, leading to heavy and slow algorithms which prohibits user interaction in real time. An example for such an application is fluid simulation, which is widely used in the film industry for creating realistic yet controllable fluid animations and for special effects where a fluid-like behavior is desired. In such simulations one often has to track the entire volume of the fluid (represented as particles or as a fixed grid), yielding a long and heavy computation. In this work, we are interested in a special case of fluid simulation where the computational cost can be reduced significantly by tracking only the boundary of the fluid.

The special phenomenon we consider in this work is *viscous fingering* [47], in which compelling patterns are generated due to instabilities at the interface of a viscous liquid. These patterns are often observed when a liquid flows into a porous medium and are related to other phenomena such as snowflake formation, bacterial growth and dendritic growth. One way to study these phenomena is to inject a less viscous liquid into a more viscous one which is trapped between two parallel plates separated by a small gap, also known as a Hele-Shaw cell. Such a setup creates a slow flow, called a Hele-Shaw flow, leading to the generation of viscous fingers which can be controlled by different parameters such as the viscosities, the surface tension at the interface of the two fluids and the rate of injection.

As these patterns has been an inspiration for artists and designers, it would be potentially useful to simulate them numerically, and allow the user to control the generated fingers formation while preserving the physical behavior and the appearance of the fluid. To this end, we devise a *boundary integral* formulation of the problem based on the special properties of the flow. Specifically, the Hele-Shaw flow is a potential flow, driven by the pressure of the fluid which is a harmonic function. For representing the harmonic function governing the flow we use the Cauchy-Green complex barycentric coordinates, which simplify the derivation and analysis of the problem, and provide an efficient way for representing the pressure by considering only values stored on the boundary of the domain. We show how these coordinates can be integrated

into a formulation of the Hele-Shaw equations, which allows us to simulate the flow at interactive rates and thus allows the user control the generated flow in real time. Additionally, we allow an injection from a line segment source and show how the Cauchy-Green coordinates can be applicable to the exterior of a planar bounded domain and multiply connected domains, which enable the generation of a large variety of patterns and different types of flows.

The second problem we consider is finding a mapping between two planar domains, which can be useful for transferring a texture between the domains, sketch based deformation and image warping. The mapping is often required to have several properties such as preservation of angles between two vectors originating at the same point (informally can be described as preservation of the texture fidelity) and low area distortion. The amount of angle distortion is called the conformal distortion, and a *conformal map* is a mapping with no angle distortion. Previous approaches for solving the problem often fix the correspondence between the boundary points of the domains and try to find a mapping for the interior points which minimizes the conformal and area distortion [48], [57]. In [22] the boundary correspondence was also updated in order to achieve a lower conformal distortion, however it requires (as well as the previous methods) a discretization of the interior of the domain. A few methods seek for a conformal map from the given domain to the unit disk, such as circle packing [50] and Schwarz Christoffel mapping [20]. However, the circle packing method involves an iterative process which slowly converges and the Schwarz Christoffel mapping requires solving a set of nonlinear equations which becomes slow for large polygons.

In our method we use the Cauchy-Green coordinates [54] web for parameterizing a space of conformal maps from the source domain, and thus compute a continuous map without requiring the discretization of the interior of the domain. We then use an iterative minimization algorithm for finding a boundary correspondence which is the closest to the boundary mapping of a conformal map represented by the coordinates. The boundary integral representation yields a very efficient method, which allows to interactively modify additional user-provided constraints for guiding the map, such as point-to-point and stroke-to-stroke correspondences. Furthermore, we show how this method can be easily generalized to quasi-conformal maps, thus enriching the space of mappings which allows to find maps with a lower area distortion.

Both of the applications described above leverage the Cauchy-Green coordinates for achieving a boundary only formulation of the problems, leading to efficient algorithms. User interaction can then be integrated into these algorithms, providing control which allows guiding the algorithms toward the desired solution.

Chapter 2

Preliminaries

2.1 Cauchy-Green coordinates

Both of the applications described in this work utilize an efficient method for representing holomorphic maps (complex differentiable functions), namely the Cauchy-Green coordinates. This representation relies on a significant result from complex analysis, the Cauchy integral formula [6], which informally states that the value of a holomorphic function at any point inside a domain can be calculated from its boundary values. More formally, given a simply connected domain Ω and some point $z \in \Omega$, the value of the holomorphic function $f : \overline{\Omega} \to \mathbb{C}$ can be calculated by

$$f(z) = \frac{1}{2\pi i} \oint_{\partial\Omega} \frac{f(w)}{w - z} dw \tag{2.1}$$

The Cauchy-Green coordinates [54] provide a discrete representation of holomorphic maps by discretizing Cauchy's integral formula. The boundary of the domain Ω is represented in the discrete case as a polygon with n vertices $S = \{z_j\}_{j=1}^n$, and the holomorphic function is discretized by storing a complex value at each vertex of the polygon $f_j = f(z_j)$, while the value of the holomorphic function on an edge is calculated by linear interpolation. Applying this discretization, we get to

$$f(z) = \frac{1}{2\pi i} \sum_{j=1}^{n} \int_{e_j} \frac{f_j + \frac{w - z_j}{z_{j+1} - z_j}(f_{j+1} - f_j)}{w - z} dw$$

We proceed by calculating the integral on each edge analytically and arranging terms together, yielding the following discrete representation

$$f(z) = \sum_{j=1}^{n} C_j(z) f_j \tag{2.2}$$

The set $\{C_j(z)\}_{j=1}^n$ are called the Cauchy-Green coordinates of the point z, and they are given by the expression

$$C_j(z) = \frac{1}{2\pi i}\left(\frac{B_{j+1}}{A_{j+1}}\log\frac{B_{j+1}}{B_j} - \frac{B_{j-1}}{A_j}\log\frac{B_j}{B_{j-1}}\right) \tag{2.3}$$

where $B_j = z_j - z$ and $A_j = z_j - z_{j-1}$ (see Figure 2.1).

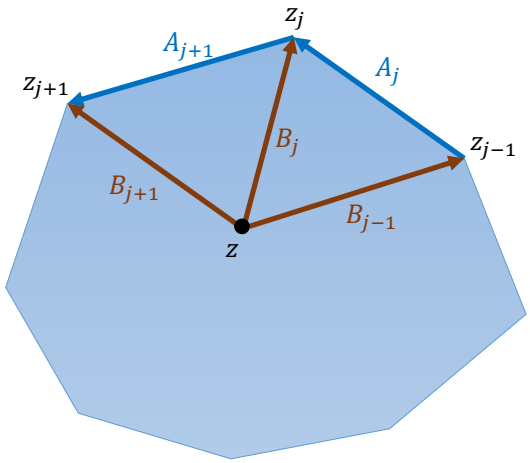


Figure 2.1: Notations for the Cauchy-Green coordinates.

Note that if the coordinates are stored in a column vector $\hat{C}(z)$ and the coefficients are stored in a column vector $\hat{f}$, then the value of $f(z)$ is given by $f(z) = \hat{C}(z)^T\hat{f}$. Thus, evaluating the function at different points $\hat{p} = \{p_r\}_{r=1}^m$ inside the domain can be achieved by packing the coordinates of the points in a matrix C of size $m \times n$ where row r contains the coordinates for point p_r, and multiplying the matrix by the vector of coefficients $\hat{f}$: $f(\hat{p}) = C\hat{f}$.

The first and second derivatives of the function is calculated by differentiating equation 2.2 with respect to z, leading to the derivative coordinates which are given by

$$D_j(z) = \frac{1}{2\pi i}\left(\frac{1}{A_{j+1}}\log\frac{B_j}{B_{j+1}} + \frac{1}{A_j}\log\frac{B_j}{B_{j-1}}\right)$$
$$D_j^{(2)}(z) = \frac{1}{2\pi i}\left(\frac{1}{B_{j-1}B_j} - \frac{1}{B_jB_{j+1}}\right)$$

Then, the first derivative of the function is given by $f'(z) = \sum_{j=1}^n D_j(z)f_j$ and the second derivative is given by a similar expression involving the second derivative coordinates.

Chapter 3

Hele-Shaw Flow Simulation with Interactive Control using Complex Barycentric Coordinates

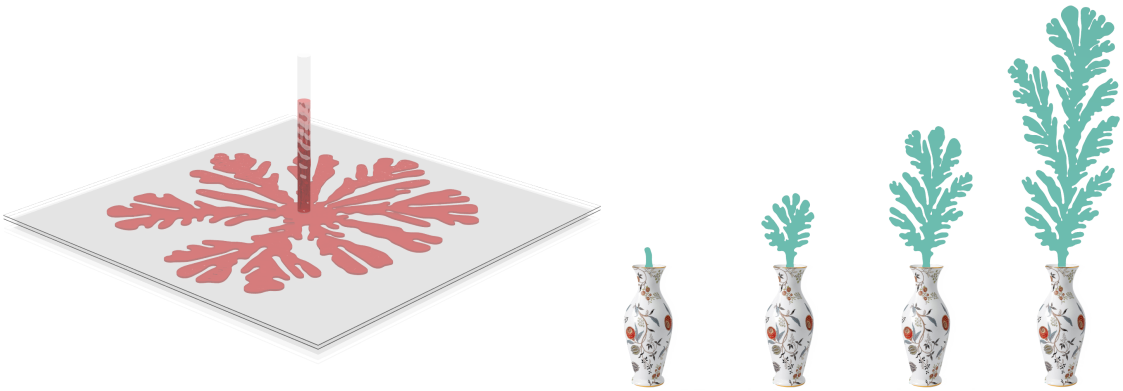


Figure 3.1: (left) A typical setup of the Hele-Shaw experiment with our simulation results. (right) One of the effects obtained by our simulation.

3.1 Background

The interaction between fluids often leads to compelling visual phenomena, such as mixing and pattern formation. In this paper we are interested in *viscous fingering*, which are the patterns generated at the unstable interface of a viscous liquid. Such patterns can arise when a liquid flows into a porous medium (e.g. sand), and are closely related to other pattern phenomena such as bacterial growth and snowflake formation. One option to experimentally study such fingering phenomena, is to inject air into a viscous liquid trapped between two parallel plates separated by a small gap (see Figure 3.2), also known as a *Hele-Shaw cell* [47]. This setup allows experimental and mathematical analysis of the pattern formation, as the governing equations for the expanding air bubble are the same as those of other more complex flows yielding similar phenomena.

From the Computer Graphics perspective, such flows generate intricate patterns

which have inspired artists [29] and designers [43]. It would therefore be potentially useful to simulate such patterns numerically, and allow the user to control the finger formation, while preserving the physical behavior and appearance of the liquid. While a plethora of methods exist for numerically simulating this phenomenon in the Computational Fluid Dynamics literature, the vast majority requires copious amounts of computational resources, and are thus not amenable to user control at interactive rates. Furthermore, traditional fluid simulation methods from Computer Graphics, such as a full Navier-Stokes simulation, is unnecessarily computationally heavy: there is no need to simulate the full behavior of the fluid in the domain, since the fingering phenomena happen at the moving free boundary.

In the spirit of recent methods for fluid simulation using boundary tracking [34], we suggest a *boundary integral* formulation for this problem. Our main observation is that the problem formulation shares many properties with the problem of *planar shape deformation*, where the behavior is prescribed by user constraints, rather than by the laws of physics. We therefore propose to leverage a reduced model successfully used for shape deformation, namely *generalized barycentric coordinates*, in order to parameterize the behavior of the flow. As Hele-Shaw flow is governed by a harmonic function, we use *complex holomorphic barycentric coordinates*, which simplify the derivation and analysis.

We show how to formulate the model equations using complex barycentric coordinates, which allows us to simulate the flow at interactive rates, and thus allows user control over the direction in which the fingers grow. By controlling the domain of injection, e.g. by injecting from a *line segment* instead of a point, we further the artist's control and enable the generation of a large variety of patterns. Finally, we show that complex holomorphic coordinates are applicable to the *exterior* of a planar bounded domain, which allows us to simulate finger formation in the case of two liquids with different viscosities, as well as for *multiply connected domains*, which allows us to simulate obstacles.

3.1.1 Related Work

While fingering in Hele-Shaw cells has not been, to the best of our knowledge, simulated in Computer Graphics, the body of work dedicated to the experimental, analytical and numerical study of this phenomenon in the Computational Fluid Dynamics (CFD) literature is vast, and a complete review is beyond our scope. We therefore focus our literature overview on putting our work in context of existing schemes, by discussing the simulation of this phenomenon in other disciplines, simulation of related phenomena in graphics, and other applications in graphics which use similar tools.

Viscous fingering in Hele-Shaw cells. An excellent review on the problem of viscous fingering in two dimensions, including the Saffman-Taylor model equations, the formulation using complex analysis and conformal maps, as well as numerical experiments, appears in [9]. A more recent mathematical treatment of the problem

using a complex analytic approach is given in [27]. Experimental investigation of this problem continues to this date, including, e.g., analyzing the dependency of the emerging pattern on the viscosity ratio in two-phase flow [11]. Numerical methods in the CFD literature are diverse, including boundary integral methods [41], yet the main focus in such disciplines is long-time evolution and the emergence of limit shapes (see e.g. the largest simulation to date [40]), as opposed to computation at interactive rates which is necessary for enabling user control. For a recent review of numerical methods for this problem in CFD, see the PhD thesis [16] and references within. Finally, it is worth noting that while the Cauchy integral formula has been used before [35] for this problem, the formulation there is quite different, as the integral there is computed numerically as opposed to our approach which uses *analytic* integrals on polygonal domains, leading to a more stable computation.

Simulation of related phenomena in Graphics. For a review of the simulation of the full Navier-Stokes equations in graphics we refer to [12]. The simulation of viscous flow using reduced dimensional methods has been proposed for viscous threads [10], viscous sheets [4] and viscous thin films on curved surfaces [3], and gap coupled solids [45]. See e.g. [52], and references within, for additional approaches to viscous fluid simulation. As opposed to these methods, we only need to simulate the behavior of the boundary curve of the fluid, and therefore face different challenges. Perhaps the phenomenon most related to our approach, is the simulation of *Laplacian growth* leading to fractal pattern formation, which is governed by similar equations. Such phenomena are exhibited for example by lichen growth, as were simulated in [51, 19] using Diffusion Limited Aggregation. In [36], a dielectric breakdown model was used for efficiently simulating lightning, whereas in [37] a hybrid algorithm was used for simulating ice formation. While all these problems are related to ours, the formulation of Hele-Shaw flow requires the use of dedicated solutions, which are both efficient and user controllable.

Other applications using similar tools. Our numerical simulation is based on complex-valued holomorphic barycentric coordinates, knowns as the *Cauchy-Green (CG) barycentric coordinates*, which were first suggested for image deformation in [54], following their initial introduction using a real-variable formulation [42]. These coordinates were later extended to allow for conformal maps with sharp bends [56], to non-holomorphic functions [55], and to three-dimensions [8]. The CG coordinates are a special case of *generalized barycentric coordinates*, which are used in graphics mostly for cage-based shape deformation, see e.g. [23], for a recent review. The CG coordinates are based on a *boundary integral formulation*, formulated in complex variables for ease of analysis, using analytical, as opposed to numerical, integration. It has been shown [54] that these coordinates are well-behaved even near the boundary of the domain, as they have a non-singular limit there, which motivates their use for the simulation of Hele-Shaw flows. Recently, beyond shape deformation, boundary element formulations have been

used in graphics for, e.g., fluid simulation [34, 13, 25], sound simulation [59], and crack simulation [28, 60].

3.1.2 Contributions

Our main contribution is a formulation for efficiently simulating Hele-Shaw flow with viscous fingering at interactive rates, while allowing for user control, using Cauchy-Green barycentric coordinates. Specifically, we:

- Formulate the model equations of the Hele-Shaw flow in terms of the Cauchy-Green coordinates, which leads to an efficient numerical simulation method (Sections 3.2, 3.3).
- Show that the Cauchy-Green coordinates are applicable to more general problems, such as exterior domains, and multiply connected domains, which allows us to simulate two-phase flow, and flow with obstacles (Section 3.4).
- Show a variety of effects that can be achieved with our technique (Section 3.5).

3.2 One phase Hele-Shaw Flow

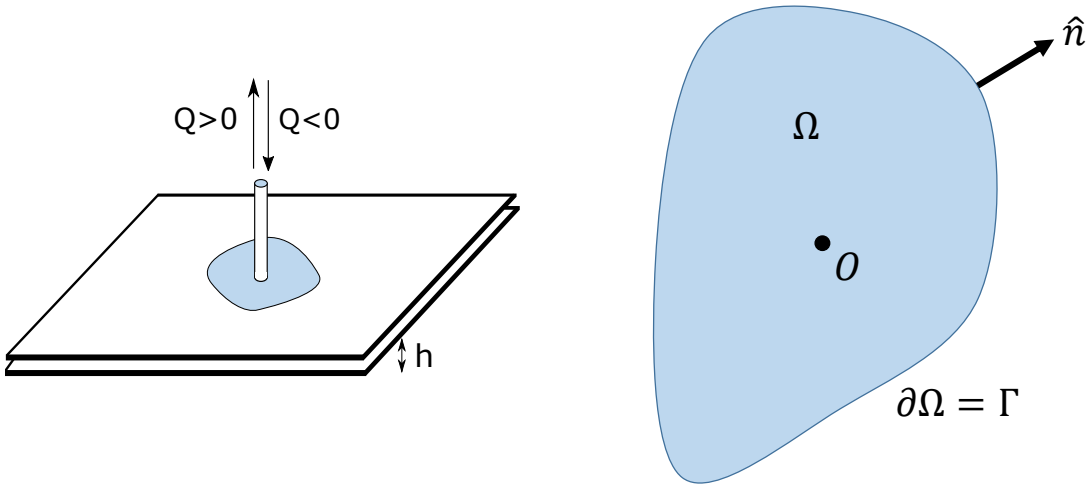
3.2.1 The Model.

The Physics. We investigate the evolution of an incompressible viscous liquid slowly injected into (or pumped out of) two parallel plates separated by a small gap, under the influence of surface tension, and without gravity. To simplify the exposition, we initially assume that the surrounding fluid is air (i.e. has zero viscosity and constant pressure), and extend later to more general settings. We further assume no-slip boundary conditions at the interface between the liquid and the plates, and a freely evolving liquid-air interface. Figure 3.2(a) illustrates this scenario.

The general Navier-Stokes equations describing fluid motion can be considerably simplified under the aforementioned assumptions. Specifically, the fluid velocity can be integrated across the gap, yielding a reduced model in terms of the two-dimensional averaged velocity V. Following the derivation presented in [27], the governing equation is $V = -\nabla\Phi$, where Φ is a scalar *potential function*, related to the physical pressure p by $\Phi = (h^2/12\mu)p$, with h the gap height and μ the fluid viscosity.

Assuming the fluid is incompressible and fills the entire gap (therefore having a constant height h) the fluid averaged velocity is divergence free everywhere except at the injection point, which we assume to be at the origin. There we have a source of strength $Q < 0$ representing a constant rate of injection. If the fluid is pumped out of the cell, Q will be positive instead. Thus, in the interior of the fluid domain we have:

$$\Delta\Phi = Q\delta_0(x, y),$$



(a) (b)

Figure 3.2: The Hele-Shaw cell model. (a) The physical model. (b) The geometry and notation.

where Δ is the Laplacian and $\delta_0(x, y)$ is the two-dimensional Dirac distribution supported at the origin.

The boundary conditions for the pressure p are given by the *Young-Laplace condition*, namely the pressure difference at the fluid-air interface is proportional to the mean curvature of the interface. Assuming constant air pressure at the exterior of the fluid, we can eliminate it from the equation by shifting both pressures by a constant factor. Furthermore, in the reduced two-dimensional model, the mean curvature of the interface is the curvature κ of the boundary curve, yielding the boundary conditions $\Phi = \sigma\kappa$, where σ is a rescaled surface tension parameter.

The Geometry. From the geometric perspective, the fluid occupies a time-dependent planar domain $\Omega(t) \subset \mathbb{C}$, which we assume to be simply-connected. Note, that we switch to complex-variable notation for points in the xy plane, namely we denote the point $(x, y) \in \mathbb{R}^2$ by $z = x + iy$, where i is the imaginary unit. The aforementioned model equations for the potential and velocity can be formulated as an evolution problem for the boundary of the domain $\Gamma(t) = \partial\Omega(t)$, given in terms of the time-varying scalar potential $\Phi(t) : \Omega(t) \to \mathbb{R}$ [27, pp. 17]:

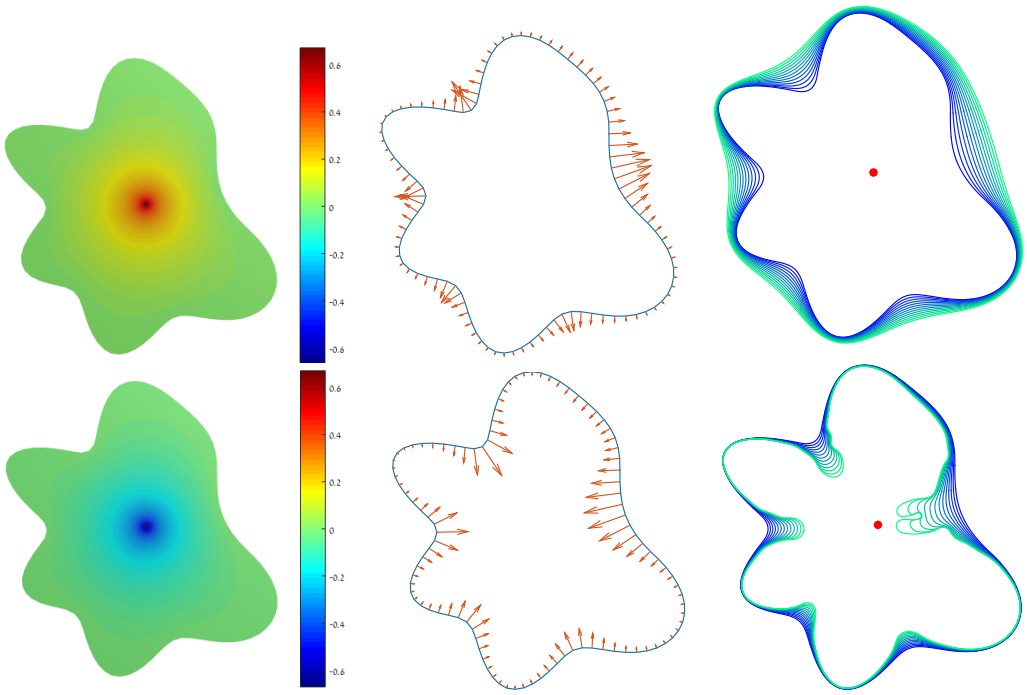
$$\Delta\Phi(z) = Q\delta_0(z), \qquad z \in \Omega \tag{3.1a}$$

$$\Phi(z) = \sigma\kappa(z), \qquad z \in \Gamma \tag{3.1b}$$

$$v_n = \langle \frac{\partial\Gamma}{\partial t}(z), \hat{n}(z)\rangle = \langle -\nabla\Phi(z), \hat{n}(z)\rangle, \qquad z \in \Gamma, \tag{3.1c}$$

where $\hat{n}$ is the outward unit normal direction of the boundary curve Γ (see Figure 3.2(b)). The first two equations yield a unique solution for the potential $\Phi(t)$, and the last equation specifies that the fluid-air interface (namely the boundary $\Gamma(t)$ of the domain) evolves according to the normal velocity $v_n = \langle V, \hat{n}\rangle$.

Given an input initial domain $\Omega(0)$, our goal is to efficiently find a family of domains $\Omega(t)$ which fulfill the model equations (3.1a)-(3.1c). To understand the behavior of the flow, consider the equations for the *zero surface tension* case (ZST), when $\sigma = 0$. In this case, the value of Φ on the boundary is 0, thus when the fluid is injected (i.e. $Q < 0$), the potential in all the domain is *positive*. Hence, the velocity at the boundary points *outward* and the boundary expands. Intuitively, points closer to the singular point at the origin will have a larger potential gradient, and therefore move faster *away* from the origin. This effect tends to *smooth* the curve. See Figure 3.3 (top) for an example showing the potential (a), the resulting velocity (b), and a few evolutions of the front under injection (c).



(a) (b) (c)

Figure 3.3: (top) injection and (bottom) suction, with zero surface tension. (a) The potential Φ at $t = 0$ is positive for injection and negative for suction. (b) Boundary velocity: points closer to the source have higher velocity. (c) Curve evolution: the curve is smoothed for injection and sharpened for suction. The original curve is shown in blue, and later iterations in green.

If, on the other hand, the fluid is pumped out (i.e. $Q > 0$), the potential is negative in all the domain, and the velocity points towards the *interior*. In this case as well points closer to the origin will move faster, but now the movement is *towards* the origin, enhancing the curvature (see Figure 3.3 (bottom)). This property makes the front unstable, as small perturbations grow, and is the cause for the fingering phenomena. Numerically, this is one of the reasons simulating this flow is challenging: a naive discretization of the model equations in the case of suction (which is the interesting case generating the pleasing visual phenomena) might quickly become unstable and cease to evolve. While the surface tension term acts as a regularizer, careful numerical

treatment is still required in order to evolve the front in a stable and efficient manner.

To do that, we leverage an important property of the system, namely that it is described by *harmonic* functions, which allows us to reformulate the problem in terms of *boundary* information only. Specifically, we will consider two approaches, modeling the behavior of Γ and Φ, respectively. In both cases, reformulating the problem in terms of complex functions is instrumental, due to the wide applicability of complex methods to the analysis of harmonic problems in two-dimensions [15].

The Complex Formulation. We briefly mention some complex analysis notation which is required for the following discussion, and refer the reader to the excellent book [1] for a thorough introduction. We slightly abuse notation, by treating planar vectors (x, y) as the complex number $x+iy$, thus for example, the gradient of a real function $\phi : \mathbb{C} \to \mathbb{R}$ corresponds to the complex number $\partial\phi/\partial x + i\partial\phi/\partial y$. A *holomorphic* function is a function that is *complex differentiable*, namely the limit $\partial f/\partial z(z_0) = lim_{z\to z_0} f(z)-f(z_0)/z-z_0$ exists regardless to the direction in which z approaches z_0.

The *Cauchy-Riemann* equations [1] formalize the relation between a holomorphic function $f(z) = \phi(z) + i\psi(z)$ and its real and imaginary parts $\phi, \psi : \mathbb{C} \to \mathbb{R}$. Specifically, ϕ, ψ are harmonic, and their gradients are orthogonal and of equal norm. Furthermore, any harmonic function is the real part of some holomorphic function. Thus, we can rephrase the Hele-Shaw model equations using a holomorphic complex potential $W : \Omega \to \mathbb{C}$, whose real part agrees with the real-valued potential: $\mathrm{Re}(W) = \Phi$.

Reformulating Equations (3.1a)-(3.1c) using W we have [27, pp.17-18]:

$$W(z) = \frac{Q}{2\pi}\log(z) + g(z), \qquad z \in \Omega \tag{3.2a}$$

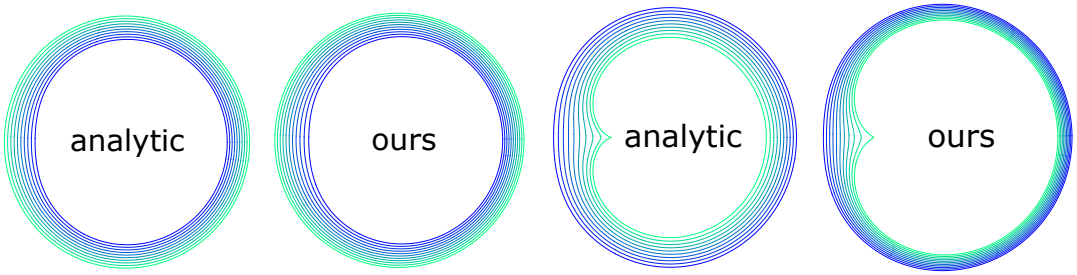
$$\mathrm{Re}(W(z)) = \sigma\kappa(z), \qquad z \in \Gamma \tag{3.2b}$$

$$v_n = -\mathrm{Re}(\frac{\partial W}{\partial z}\hat{n}(z)), \qquad z \in \Gamma, \tag{3.2c}$$

where g is a holomorphic *regular* function (i.e. without poles in Ω). For the first equation we used the fact that $\mathrm{Re}(1/2\pi \log(z)) = 1/2\pi \log(|z|)$ is the Green's function for the Laplacian in the plane, and thus solves Equation (3.1a), whereas g is used to fulfill the boundary conditions (3.1b). Finally, the third equation is due to the representation of the inner product of two planar vectors in complex form: $\langle a, b\rangle = \mathrm{Re}(\overline{a}b)$, and the relation between the derivative of a holomorphic function and the gradient of its real part, yielding: $\partial W/\partial z = \partial\Phi/\partial x - i\partial\Phi/\partial y$.

With the complex formulation at hand, we can now attempt to address the model equations. We will propose two approaches, with complementary advantages. First, we will leverage the invariance of harmonic functions under *conformal* (angle preserving) maps, to directly solve for the evolving front $\Gamma(t)$ by parameterizing it as a time-evolving conformal map (and thus a holomorphic function) from the unit disk. This allows us to handle both injection and suction, and produces similar behavior as a known analytic solution for the challenging case of suction with zero surface tension. Unfortunately, this

approach is difficult to extend to more general scenarios (e.g. non-zero surface tension and two-phase flow), and causes additional technical problems due to uneven sampling of the evolving front. Our second approach is to directly solve for the evolving complex potential W, and it can be applied in a variety of scenarios, yet cannot reproduce the analytic solution. Still, this approach is highly useful in practice, as it is easily modified to allow for user control, and is efficient enough to allow interactivity. Note that Figure 3.4 was produced with the first approach, and all the others were produced with the second approach.



(a) (b) (c) (d)

Figure 3.4: Comparison of the quadratic form analytic approach for injection (a) and suction (c) with our approach for injection (b) and suction (d), using the same initial curve $\Omega(0)$. Note that our method indeed produces a cusp similar to the cusp of the analytic solution.

3.2.2 Evolving the Interface

The Riemann Mapping Theorem states that for any simply connected domain $\Omega \subset \mathbb{C}$ there exists a unique bijective conformal mapping which maps the unit disk $U = \{\zeta : |\zeta| < 1\}$ into Ω such that $f : U \to \Omega, f(0) = 0, f'(0) \in \mathbb{R}^+$. Thus, we can track the time-varying domain of the fluid $\Omega(t)$ by the time-varying conformal map $f(\zeta, t)$ from the unit disk into $\Omega(t)$ for every t (see Figure 3.5).

The *Polubarinova-Galin (PG) equation* [27] provides a condition that the conformal mapping $f(\zeta, t)$ must satisfy (in the case of a single singular point at the origin ($s = 0$) and zero surface tension) for the model equations to hold. It builds on three facts: First, harmonic functions are invariant under conformal maps, and thus given a solution to the model equations on U we can use the conformal map to get a solution on Ω. Second, the normal velocity v_n can be expressed both in terms of the complex potential W_Ω and the time derivative of the conformal map $\partial f/\partial t$. And finally, the normal $\hat{n}$ on Ω can also be computed using f (as seen in Figure 3.5).

Combining these facts yields the equation (see supplemental material) [27, Eq. (1.16)]:

$$\mathrm{Re}\left(\frac{\partial f}{\partial t}\zeta\overline{\frac{\partial f}{\partial \zeta}}\right) = -\frac{Q}{2\pi}, \quad \zeta \in \partial U. \tag{3.3}$$

It was shown [26] that in the case of injection under some assumptions on smoothness of $\partial\Omega(0)$ there exists a unique solution $f(\zeta, t)$ satisfying the PG equation. It is also

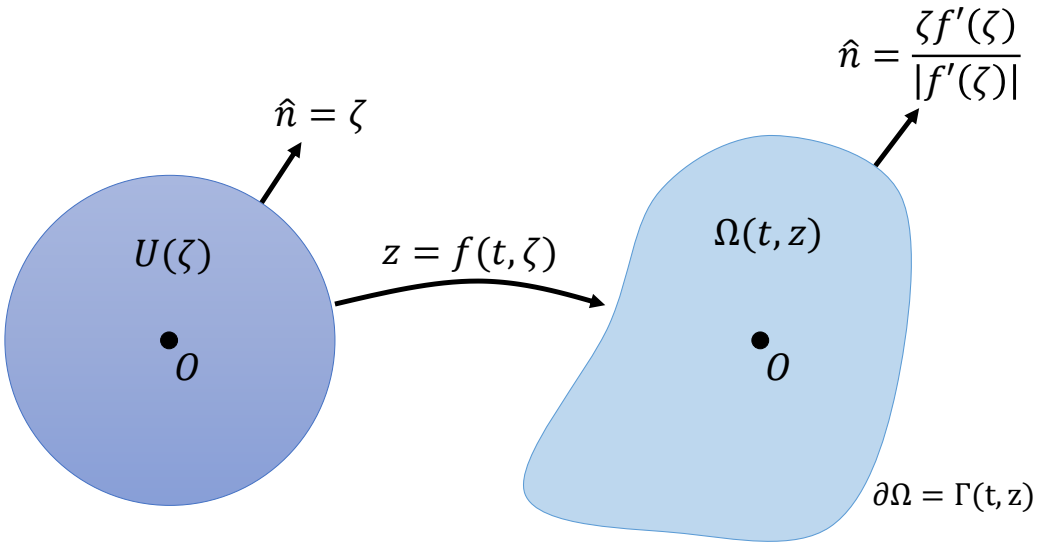


Figure 3.5: Notation for evolving the interface. We map the unit disk $U(\zeta)$ (left) using a time varying conformal map $f(\zeta, t)$ to a time-varying domain $\Omega(t, z)$ with boundary $\Gamma(t, z)$ (right). The normal to the disk is mapped with the derivative of the map f' to the scaled normal at the target domain.

possible to find analytic solutions by using a special form for $f(\zeta, t)$ (i.e. expressing specific types of boundaries). For example, in [24] the author chose the quadratic form $f(\zeta, t) = a_1(t)\zeta + a_2(t)\zeta^2$ where $a_1(t)$ and $a_2(t)$ are real coefficients. Substituting $f(\zeta, t)$ into (3.3) gives two equations which can be solved for the coefficients a_1, a_2 at time t, yielding an explicit solution for the problem.

In the next section we discuss the spatial discretization using the Cauchy-Green barycentric coordinates for this formulation, and the resulting discrete equations. Figure 3.4 shows such solutions to the PG equation for injection and suction, using the quadratic form approach and our approach, using the same initial curve $\Omega(0)$. Note, that our method produces similar behavior to the analytic solution.

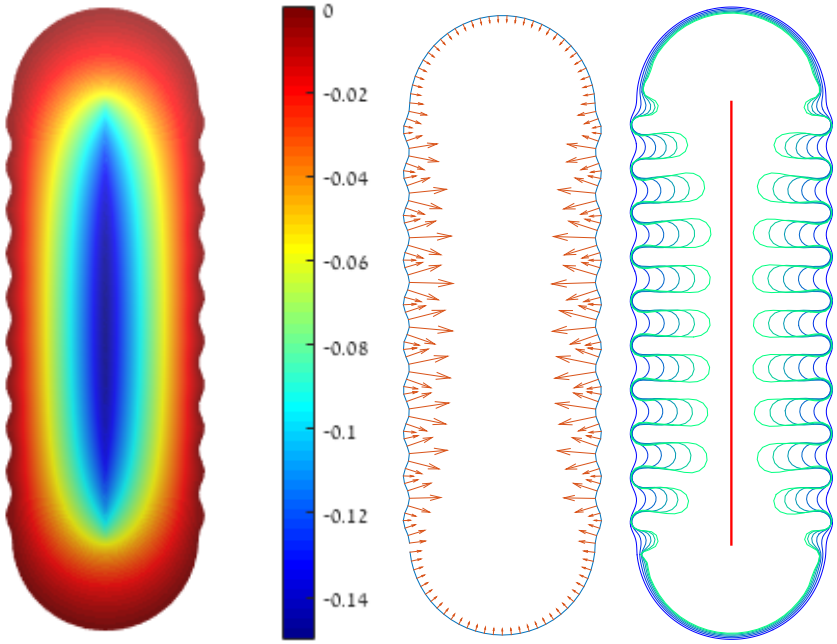
3.2.3 Evolving the Potential

As solving for the conformal map f has several issues, we alternatively suggest to find the complex potential $W(z)$ which satisfies Equations (3.2a)-(3.2c). We do so by solving for the holomorphic function $g(z) : \Omega \to \mathbb{C}$, which satisfies the boundary conditions: $\mathrm{Re}(g(z)) = -Q/2\pi \log|z| + \sigma\kappa(z)$. Interestingly, holomorphic functions and conformal maps are equivalent, thus we can use the same ansatz for the spatial discretization, namely the discrete Cauchy-Green coordinates. Furthermore, this approach is more easily generalizable to handle multiple singularities of different types.

Mutliple Singularities. In the physical model, extending to multiple singularities implies that instead of having a single source or sink of the velocity at the origin, there are multiple sources and sinks at locations $s_k \in \Omega$, with strengths Q_k. Thus Equation (3.1a)

changes to $\Delta\Phi = \sum_k Q_k \delta(z - s_k)$. Since Green's functions can be superimposed, the corresponding contribution to the complex potential is $\sum_k {}^1\!/_{2\pi} Q_k \log(z - s_k) = \sum_k W_s(z, s_k)$.

Similar reasoning allows us to add *line singularities*, namely sources and sinks which are localized on line segments. Given a line segment $l : s(t) = z_1 + t(z_2 - z_1)$, its contribution to the complex potential is $W_l(z, l) = {}^1\!/_{2\pi} Q_l \int_0^1 \log(z - s(t))dt$ (see the supplemental material for the closed form solution of this integral). Fig. 3.6 shows the scalar potential and the velocities for a source localized on a line segment. Note that, compared to a point source, there is a larger neighborhood of points on the evolving curve with large velocities, yielding a more noticeable effect during the evolution.



(a) (b) (c)

Figure 3.6: Simulating a sink localized on a line segment. (a) The scalar potential Φ. (b) The velocity of the interface, note the larger region of high velocities. (c) The resulting evolution of the front.

Combining the contributions from all the singularities yields to the following modification to Equation (3.2a):

$$W(z) = \sum\nolimits_k W_s(z, s_k) + \sum\nolimits_k W_l(z, l_k) + g(z), \tag{3.4}$$

where $\{s_k\}$ and $\{l_k\}$ are the sets of point sources and line segments, respectively.

In the next section we show how the Cauchy-Green barycentric coordinates can be used for this formulation. Figure 3.7 shows an example of a flow where the point location of the singularity (i.e. the source s) changes during the flow, which allows fine control on the behavior of the fingers. Since the computation is done at interactive rates, the user can move this location interactively, yielding an intuitive tool for generating

finger-like effects (see the video).

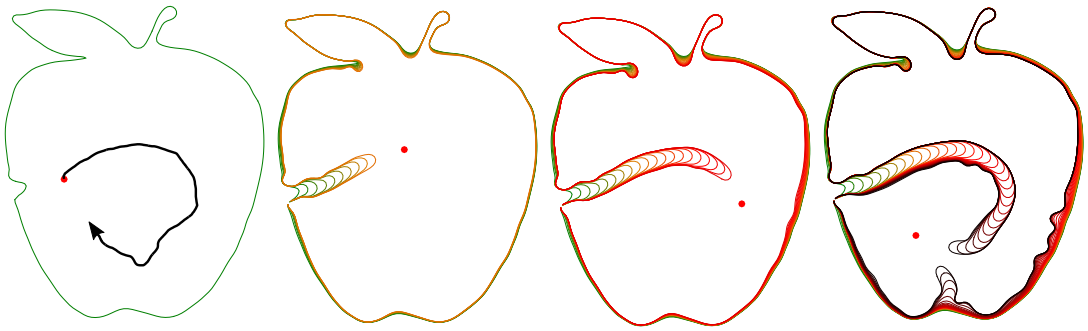


Figure 3.7: A few frames from an interactive simulation, where the user modifies the singularity's location in real-time. The resulting singularity path is shown on the left. Note how the path of the fingers is modified to "aim" for the location of the closest singularity.

3.3 Discretization

In the previous section we described how the model equations of Hele-Shaw flow can be reduced to finding a time-varying holomorphic function, representing either a conformal map from the unit disk to the fluid domain, or the regular part of the complex potential of the fluid domain, under some constraints. This setup is remarkably similar to the setup common in *planar shape deformation*, where we seek a deformation of the input shape which is *detail preserving*, under some user constraints. In [54] it was proposed to use the machinery of conformal maps for this problem, yielding exactly the same mathematical formulation as we have, namely, finding a time varying conformal map under some constraints. We now leverage that machinery to get a deformation which is conformal, yet driven additionally by the physical model, rather than exclusively by a human user.

3.3.1 Cauchy-Green Coordinates.

The *Cauchy integral formula* [6] is a central result in complex analysis, expressing the fact that the values of any holomorphic function inside a domain Ω can be calculated by the following integral on the boundary of Ω:

$$f(z) = \frac{1}{2\pi i} \oint_{\partial\Omega} \frac{f(w)}{w - z} dw, \quad z \in \Omega. \tag{3.5}$$

The Cauchy-Green Coordinates [54] are a discretization of the Cauchy integral. The domain Ω is discretized using a polygon on which we store the function as values at the vertices $\{f_j\}_{j=1}^n$. The function $f(w)$ is approximated on each edge by a linear interpolation between these values. Then, the integration on the edges can be calculated analytically, yielding a complex coefficient $C_j(z)$ for each f_j. These complex

coefficients are called the *Cauchy-Green barycentric coordinates*. Finally, the integral is approximated using the sum:

$$f(z) = \sum_{j=1}^{n} C_j(z) f_j.$$

Similarly, the derivative of f can be approximated using the derivative of $C_j(z)$:

$$f'(z) = \sum_{j=1}^{n} C_j'(z) f_j = \sum_{j=1}^{n} D_j(z) f_j.$$

We provide the expression for the Cauchy-Green coordinates and their gradients in the supplemental material. In the following we show how the CG coordinates can be used for evolving the interface and the complex potential.

3.3.2 Evolving the Interface

Spatial Discretization. We search for a time varying conformal map $f : U \to \Omega$, which satisfies Equation (3.3). We discretize the unit circle using a regular n-sided polygon $\hat{U}$, and represent the conformal map using n functions $f_j(t), j \in 1..n, t \in \mathbb{R}$, corresponding to the vertices of the polygon. Then, the map of $\hat{U}$ is:

$$f(\zeta, t) = \sum_{j=1}^{n} C_j(\zeta) f_j(t), \quad \zeta \in \hat{U}. \tag{3.6}$$

Since $C_j(\zeta)$ are independent of f, the time derivative is given by: $\frac{\partial}{\partial t} f(\zeta, t) = \sum_j C_j(\zeta) \frac{\partial}{\partial t} f_j$. Thus, the semi-discrete PG equation corresponding to Equation (3.3) is, for $\zeta \in \partial\hat{U}$:

$$\mathrm{Re}\left(\left(\sum_{j=1}^{n} C_j(\zeta) \frac{\partial f_j}{\partial t} \right) \overline{\left(\zeta \sum_{m=1}^{n} D_m(\zeta) f_m \right)} \right) = -\frac{Q}{2\pi}. \tag{3.7}$$

We additionally sample the regular polygon at the points $S = \{\zeta_l\} \in \partial\hat{U}$, which leads to the space-discrete system of ODEs:

$$\mathrm{Re}\left((\mathcal{C} \frac{\partial}{\partial t} \hat{f})_l \overline{(\mathcal{D}\hat{f})_l} \right) = -\frac{Q}{2\pi}, \forall l \in 1..|S|, \tag{3.8}$$

where $\mathcal{C}, \mathcal{D}$ are complex matrices with entries $C_{lj} = C_j(\zeta_l)$ and $D_{lm} = \zeta_l D_m(\zeta_l)$, respectively, and $\hat{f}$ is a vector with entries $f_j(t)$.

Time Discretization. We use an explicit Euler scheme to integrate equation (3.8). Specifically, given $\hat{f}^k$ at iteration k, we find a discrete approximation of $\frac{\partial}{\partial t}\hat{f}^k$, denoted by $(\Delta\hat{f})^k$, by minimizing the error of an over-constrained set of linear equations derived from Equation (3.8) sampled at $4n$ points. Finally, we set $\hat{f}^{k+1} = \hat{f}^k + \Delta t (\Delta\hat{f})^k$ for a constant delta time $\Delta t = 0.001$. Figure 3.4 (top) shows a comparison of our evolution for the case of injection with the classic solution obtained using the quadratic complex form, where we achieve similar behaviour.

Regularization. While this approach works for the injection problem, the suction problem requires additional regularization because of its ill-posed nature. The regularization we propose is the minimization of the second spatial derivative of $\Delta\hat{f}$, in order to keep the conformal map smooth. Thus, we add a regularization term $\lambda\mathcal{C}^{(2)}$ to the linear equations, where $\mathcal{C}^{(2)}$ are the second spatial derivatives of the Cauchy-Green coordinates in matrix form (provided in the supplemental material). Figure 3.4 shows our result with this regularization (where we used $\lambda = 0.001$) compared with the classic analytic solution. Note that we manage to achieve the characteristic cusp despite our use of regularization.

3.3.3 Evolving the Potential

Spatial Discretization. We search for a holomorphic function $g(z) : \Omega(t) \to \mathbb{C}$, given by equations (3.4) and (3.2b). We first discretize the input domain $\Omega(t)$ using n samples, to get the closed polygon $\hat{\Omega}(t)$, and then we use again the Cauchy-Green coordinates to represent $g(z)$:

$$g(z) = \sum\nolimits_{j=1}^{n} C_j(z) g_j, \quad z \in \hat{\Omega}(t). \tag{3.9}$$

The boundary conditions (3.2b) yield the constraints:

$$\operatorname{Re}\left(\sum_{j=1}^{n} C_j(z) g_j\right) = -\operatorname{Re}(W_{src}(z)) + \sigma\kappa(z), \quad z \in \partial\hat{\Omega},$$

where $W_{src}(z)$ is the combined potential of all the sources and sinks, as given in Equation (3.4).

We sample the boundary of the discrete domain at the points $S = \{z_l\} \in \partial\hat{\Omega}$, which again leads to an over-constrained system of linear equations, which can be solved for $\hat{g}$, the complex vector with entries g_j. The spatial derivative of the complex potential $\partial/\partial z W$ is computed using the known derivative of the potential at the singularities and $g'(z_l) = \sum_j D_j(z_l) g_j$. Finally, from Equation (3.2c), the normal velocity is given by $v_n = -\operatorname{Re}(\partial/\partial z W \hat{n})$, where $\hat{n}$ is the averaged normal at the vertices of $\hat{\Omega}$.

Time Discretization. We use explicit Euler integration of Equation (3.2c), and advance the sampled locations z_j using $z_j^{k+1} = z_j^k + (\Delta t) v_n(z_j)\hat{n}(z_j)$. Since in this setup we can directly prescribe non-zero surface tension σ, no regularization is required. We do, however, resample the curve $\partial\hat{\Omega}(t)$ during the evolution, taking into account the curvature. See section 3.5.1 for the details, as well as for the computation method of the dynamic time-step Δt.

While it is possible to use a more advanced time integrator, we have observed that this approach is efficient and stable. Specifically, for a constant rate of injection Q, the area of the domain should grow linearly. Figure 3.8 shows the result of injection (left) and suction (middle) from a single source using the complex potential approach and the corresponding graph denoting the change in the area (right). Note that we get a

linear change in the area, as expected.

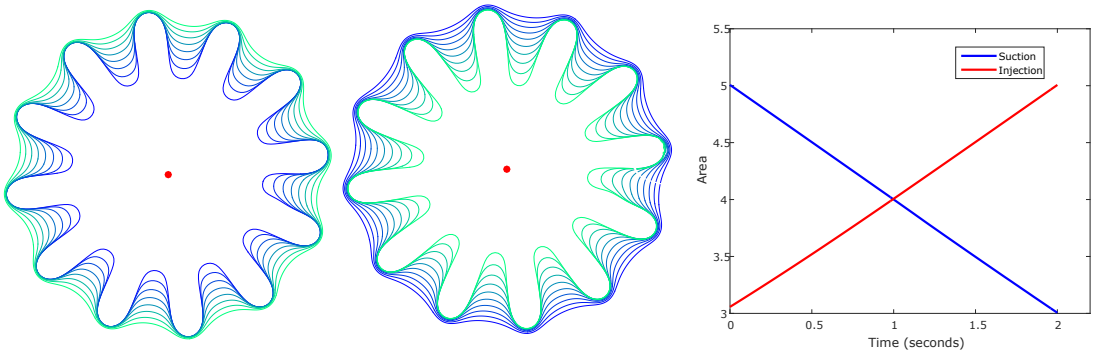


Figure 3.8: Front evolution for the stable case of injection (left) and suction (middle) with small surface tension (10^{-5}), using the complex potential approach (Section 3.3.3). The method yields linear evolution of the area as expected (right).

3.4 Extensions to the Model

The setup we presented, namely: simulating the one-phase Hele-Shaw flow by evolving the complex potential with the Cauchy-Green coordinates, can be easily extended to more complicated physical setups. We first present the generalization to *exterior flow*, namely the fluid occupies an unbounded domain in the plane which is the complement of a simply connected curve, by showing how the Cauchy-Green coordinates can be modified to handle holomorphic functions on unbounded domains. Then, by combining interior and exterior flows, we address *two-phase flow* by solving for two potential functions. Finally, we show how to handle *obstacles* using multiply connected domains and different boundary conditions.

3.4.1 One Phase Hele-Shaw Flow with a Bubble

We consider the inner fluid to be air with zero viscosity (forming a bubble inside the outer fluid) and suction or injection of the external fluid from infinity (see Figure 3.9). The flow is driven by the potential of the external fluid Φ which is related to the fluid pressure by a constant scaling factor. Since the singular point is at infinity the potential should be harmonic everywhere but behave at infinity as [17]:

$$\Phi(z) \approx -\frac{Q}{2\pi} \log|z|, \quad \text{as} \quad |z| \to \infty.$$

We therefore represent the potential as $\Phi(z) = -Q/2\pi \log|z| + h(z)$ where $h(z)$ is a harmonic function which tends to a constant at infinity, and its gradient tends to zero.

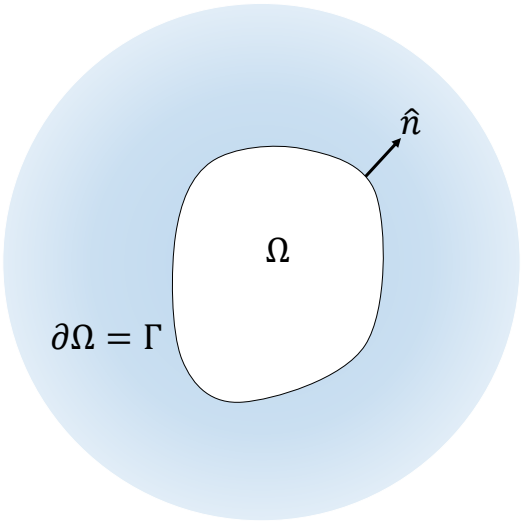


Figure 3.9: The geometry and notations of the exterior one phase flow.

The corresponding model equations are therefore:

$$W(z) = -\frac{Q}{2\pi}\log(z) + g(z), \qquad z \notin \Omega \tag{3.10a}$$

$$\mathrm{Re}(W(z)) = -\sigma\kappa(z), \qquad z \in \Gamma \tag{3.10b}$$

$$v_n = -\,\mathrm{Re}(\frac{\partial W}{\partial z}\hat{n}(z)), \qquad z \in \Gamma \tag{3.10c}$$

where $g(z)$ is a holomorphic function which satisfies $\lim_{|z|\to\infty} g(z) = const$, and $\hat{n}$ still points outward of the curve. As previously, the boundary conditions are given by the Young-Laplace condition relating the pressure difference to the curvature of the boundary. The viscosity of the inner fluid is negligible compared to the viscosity of the outer fluid, and thus the pressure of the inner fluid is assumed to be constant, leading to the boundary conditions in Eq. (3.10b). Similarly to the interior flow, we will represent the holomorphic function $g(z)$ using the Cauchy-Green coordinates, by slightly modifying them to handle exterior domains.

Exterior Cauchy-Green Coordinates. Given a bounded simply connected domain Ω and a function $f(z)$ which is holomorphic in the exterior of Ω such that $\lim_{z\to\infty} f(z) = c$ for some constant c, the following holds [33, pp. 140]:

$$\frac{1}{2\pi i}\int_{\partial\Omega}\frac{f(w)}{w-z}dw = \begin{cases} c & z \in \Omega \\ c - f(z) & z \notin \Omega \end{cases}.$$

This result is sometimes known as *Cauchy's integral formula for an unbounded domain.* Thus, we can pick an arbitrary point $a \in \Omega$, and then the value of $f(z)$ for a point $z \notin \Omega$ is given by:

$$f(z) = \frac{1}{2\pi i}\int_{\partial\Omega}\frac{f(w)}{w-a}dw - \frac{1}{2\pi i}\int_{\partial\Omega}\frac{f(w)}{w-z}dw,$$

and is independent of the choice of a.

We discretize the domain using a polygon $\hat{\Omega}$, and use the Cauchy-Green coordinates for discretizing the Cauchy integral:

$$f(z) = \sum_{j=1}^{n} C_j^e(z) f_j, \quad C_j^e(z) := C_j(a) - C_j(z), \quad z \notin \hat{\Omega},$$

where $a \in \hat{\Omega}$ is arbitrary, and $C_j^e(z)$ is the *exterior Cauchy-Green coordinate* for a vertex j of $\hat{\Omega}$. This result indicates that the exterior coordinates can be expressed using the regular Cauchy-Green coordinates, and so do their derivative as $D_j^e(z) = -D_j(z)$.

Exterior Flow. Using the exterior coordinates, we can apply our previous ansatz and discretize Equations (3.10a)-(3.10c). Specifically, we assume that $\lim_{|z|\to\infty} g(z) = const$ and represent it by $g(z) = \sum_j C_j^e(z) g_j$. As before, the discrete values g_j are calculated by solving the over-constrained linear system obtained by sampling the boundary and applying the boundary conditions $\mathrm{Re}(g(z)) = -\sigma\kappa(z) + Q/2\pi \log|z|$. Given the values of g_j, the velocity is calculated using the derivative of the exterior coordinates and the interface is advanced according to the normal velocity.

Figure 3.10 shows an example of using the exterior flow to "continue" a real Hele-Shaw flow. Specifically, we extracted from a photograph by the artist Antony Hall [29] the boundary curve of a real Hele-Shaw flow, and used it as the initial conditions of our simulation. The Figure shows the original photograph (left), and our "simulated" photograph (right), after allowing the front to evolve (the initial fluid has darker color). Note that the simulated front closely resembles the original photograph. Fig. 3.1, 3.13, 3.16 and the attached video show additional results using the exterior flow.

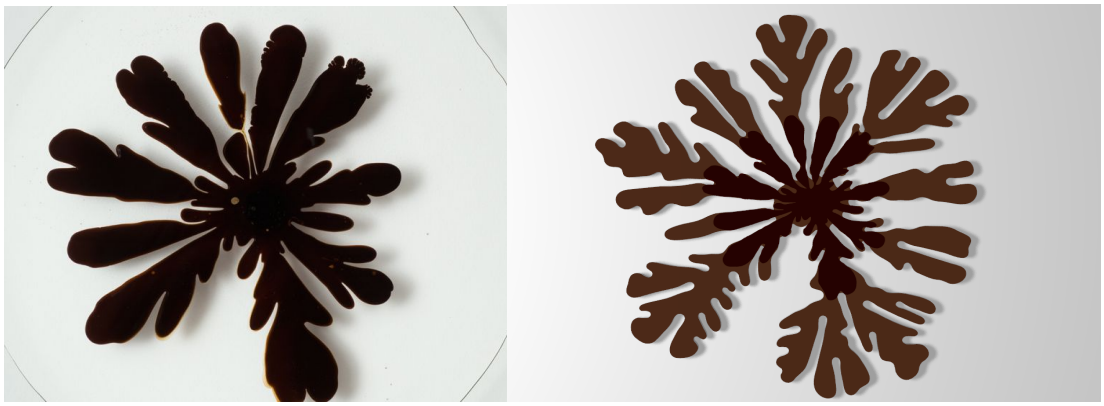


Figure 3.10: "Continuing" an experimental Hele-Shaw flow. (left) Photograph by Antony Hall. (right) Our evolution starting from the boundary curve of the photograph.

3.4.2 Two Phase Hele-Shaw Flow

In the general case, there are two fluids with non-zero viscosities μ_1 and μ_2 occupying the interior and exterior of the domain [32]. Their flow is driven by two harmonic

potentials which we denote by Φ_1 and Φ_2 for the inner and outer fluid, respectively. We again represent the potentials as the real parts of complex holomorphic potentials W_1 and W_2. For simplicity we assume a single source inside the inner fluid located at the origin. Since the fluids are incompressible, the total amount of material must not change and thus injection of material at some location must be compensated by removal of material from another. Therefore, the outer fluid will also have a singularity, and we assume it is at infinity.

The corresponding equations for this model are [32]:

$$W_1(z) = \frac{Q_1}{2\pi}\log(z) + g(z), \qquad z \in \Omega \tag{3.11a}$$

$$W_2(z) = -\frac{Q_2}{2\pi}\log(z) + h(z), \qquad z \notin \Omega \tag{3.11b}$$

$$\mu_1 \operatorname{Re}(W_1(z)) - \mu_2 \operatorname{Re}(W_2(z)) = \sigma\kappa \qquad z \in \Gamma \tag{3.11c}$$

$$-v_n = \operatorname{Re}\left(\frac{\partial W_1}{\partial z}\hat{n}(z)\right) = \operatorname{Re}\left(\frac{\partial W_2}{\partial z}\hat{n}(z)\right) \qquad z \in \Gamma \tag{3.11d}$$

where Q_1 is the strength of the singularity at the origin, Q_2 is the strength of the singularity at infinity, $g(z)$ is a holomorphic function inside Ω and $h(z)$ is a holomorphic function in the exterior of Ω which tends to a constant at infinity. Note that in order to preserve the incompressibility of the fluids we must have that $Q_1 = -Q_2$ (the rate of injection matches the rate of pumping). The holomorphic functions $g(z)$ and $h(z)$ are determined by the Young-Laplace boundary condition (3.11c), expressing the pressure jump across the interface, and the kinematic boundary condition (3.11d), stating that the normal velocities of the two fluids at the interface must be equal (as the fluids do not mix).

As before, we represent the holomorphic functions $g(z)$ and $h(z)$ using the interior and exterior Cauchy-Green coordinates $g(z) = \sum_j C_j(z)g_j$ and $h(z) = \sum_j C_j^e(z)h_j$, and discretize equations (3.11a)-(3.11d) in the same way. The coefficients g_j and h_j are found as the solution of an overconstrained linear system obtained by sampling the boundary, and the values of g_j are used for calculating the normal velocity and advance the boundary of the curve.

Figure 3.11 shows examples of two-phase flows, for the stable case of injection when $\mu_1 > \mu_2$ (bottom), and the unstable case of injection for two viscosity ratios μ_1/μ_2 (top,middle). Note that the smaller viscosity ratio generates more intricate and thinner fingers, as expected [11]. Furthermore, it is worth noting that in the extreme limits of the viscosity ratio the two previous cases are recovered. Specifically, when $\mu_2/\mu_1 \to 0$ the flow behaves as the one phase flow of the inner fluid and when $\mu_1/\mu_2 \to 0$ it behaves as the one phase flow with a bubble. Figure 3.14 and the attached video show additional results of the two phase flow.

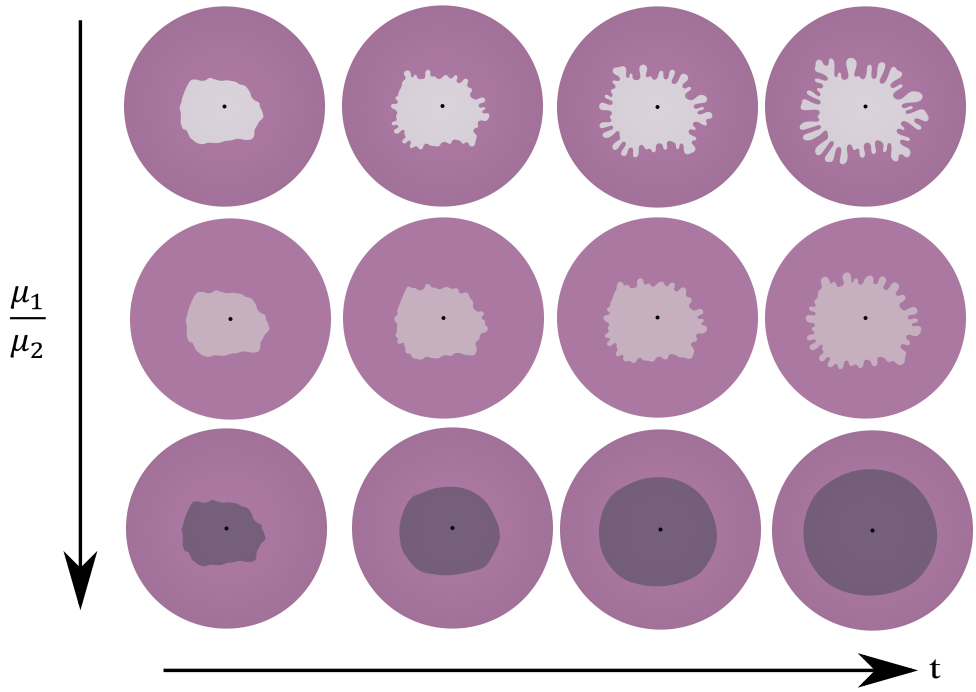


Figure 3.11: Two phase flow simulation. (top) Unstable injection with $\mu_1/\mu_2 = \mu_a = 0.01$. (middle) Unstable injection with $\mu_1/\mu_2 = \mu_b = 0.3$. (bottom) Stable injection with $\mu_1/\mu_2 = 2$. Note that the lower viscosity ratio μ_a (top) generates thinner and more intricate fingers.

3.4.3 Obstacles.

Obstacles are formulated using the *no-penetration* Neumann boundary conditions, i.e. the normal velocity of the interface along the obstacle should be zero. Here the fluid domain may be multiply-connected, and its boundary $\partial\Omega$ is composed of a free boundary denoted by Γ_1, and a part which is allowed to move only in the tangent direction (where the boundary is part of an obstacle), denoted by Γ_2. Thus, the formulation is similar to the formulation of the regular Hele-Shaw flow, with the exception that now the boundary condition for the potential function on Γ_2 is $\mathrm{Re}(\frac{\partial W}{\partial z}\hat{n}) = 0$.

Since obstacles form holes in the domain, the domain is now *multiply-connected*. Interestingly, the Cauchy integral formula holds in this case as well [6], with the modification that the orientation of the interior boundaries should be opposite to those of the exterior boundaries. Thus, we can use the same discretization as before using the Cauchy-Green coordinates to represent the regular part of the complex potential, and add the boundary conditions:

$$\mathrm{Re}\left(\left(\frac{Q}{2\pi z} + \sum_{j=1}^{n} D_j(z) g_j\right)\hat{n}\right) = 0 \qquad\qquad z \in \Gamma_2.$$

Note, that in this case $D(z)$ discretizes the multiply connected Cauchy integral. Given the fluid interface with the Cauchy-Green coordinates $D^{\Gamma}(z)$ and m holes with the

Cauchy-Green coordinates $\{D^k(z)\}_{k=1}^m$ the multiply connected coordinates are given by $D(z) = D^\Gamma(z) - \sum_{k=1}^m D^k(z)$. Fig. 3.12 shows suction from a line source in an interior flow with obstacles and Fig. 3.16 shows an external flow with obstacles.

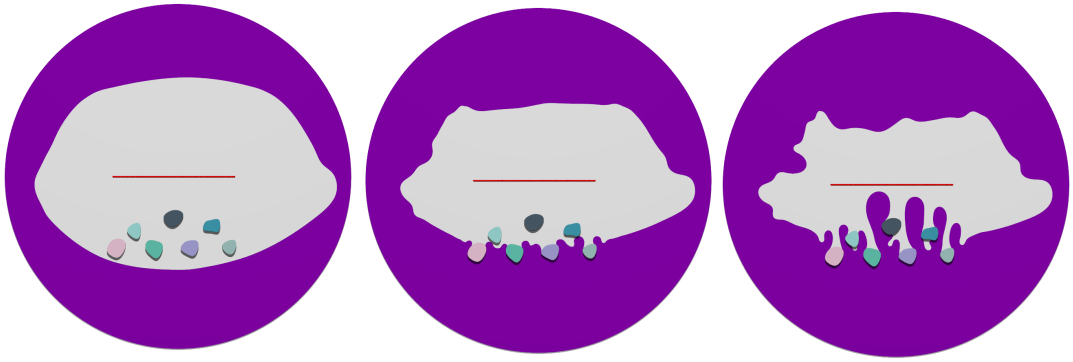


Figure 3.12: Flow with obstacles, suction from an interior segment.

3.5 Experimental Results

3.5.1 Implementation details.

User Interface. We implemented our method in MATLAB. The interface is represented as a polygon with n vertices, where n may change during the flow. The user draws a control polygon, which is then interpolated using a cubic spline and sampled at $n = 100$ points for getting the initial polygonal interface. The user adds singularity points and line singularities and chooses their strength Q. The user is free to move the singularity locations during the simulation, and thus change the direction which the fingers will follow. Using the line singularities the user can choose the path of a finger when it reaches the line (see Figure 3.13).

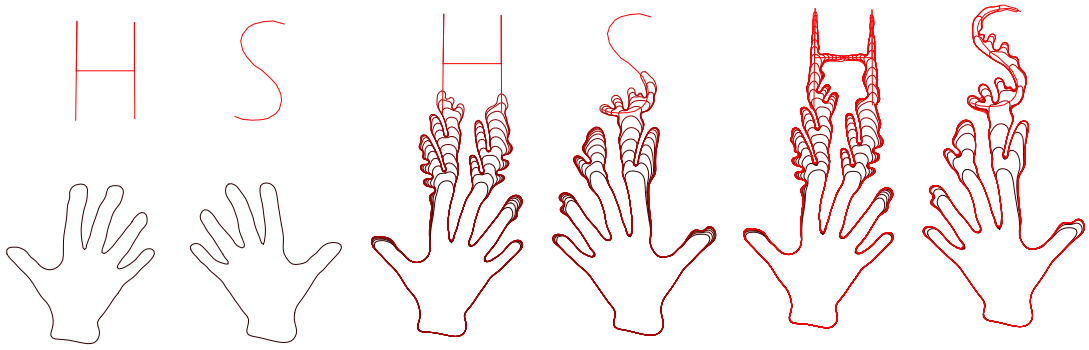


Figure 3.13: By prescribing a line singularity the user controls the path of the fingers, as they follow the line when it is reached.

Simulation. For each simulation frame the interface is sampled at $4n$ points (each edge is sampled 4 times) on which the boundary conditions are applied to obtain $4n$ linear equations. The calculation of the coordinates for these samples can be easily

parallelized, and is thus done on the GPU (on an NVIDIA GTX 980 card). To give a feel for the timings involved, calculating the coordinates of 4000 points takes 5 milliseconds. The system of linear equations is then solved by minimizing the least squares error, resulting in a vector of coefficients representing the potential. The normal velocity at each vertex is then calculated using the derivative of the coordinates, and the vertices are moved using an explicit Euler scheme with a dynamic time step, which is chosen according to the ratio of the edge length and the normal velocity $\Delta t = \min\left(|e_i|/v_n\right)$. Finally, we fit a cubic spline which interpolates the new polygon and sample it according to the curvature (i.e. more samples in the highly curved regions). The number of sampled points is chosen dynamically according to a minimal edge limit and a limit on the number of points, where we used 0.02 and 1000, respectively.

Singular integrals at the boundary. The Cauchy-Green coordinates and their derivatives can be singular when evaluated at the boundary of the domain. The coordinates, though, have a non-singular limit, given in [54], which we use for our computations. The derivatives have a non-singular limit on the edges of the boundary polygon, yet are undefined at the vertices. Thus, we calculate $\partial W/\partial z$ at a point close to the vertex inside the domain. We chose to calculate the derivative at a point with distance of 10^{-3} from the vertex in the normal direction into the interior or the exterior of the domain, depending on where the complex potential is defined (the interior or the exterior flow). In the two phase case we can calculate the velocity from either the interior or the exterior potentials. The normal of the vertices is calculated as a weighted average of the incident edges normals.

Degrees of freedom. Since the coordinates sum to one, their imaginary parts sum to zero. Thus, we have one degree of freedom which can be fixed by choosing the imaginary part of the first coordinate to be zero. In the two phase case we have three degrees of freedom: two of them are expressed as a constant addition to the imaginary parts of each of the potentials, and fixed similarly. The third is due to the Young-Laplace boundary condition, as it involves the difference between the two potentials. It is fixed by choosing the real part of the first coordinate of one of the potentials to be zero.

3.5.2 Limitations.

Our method has a few limitations. First, we do not handle topology changes, which sometimes may be required (e.g. merging fronts after passing an obstacle, or bubbles created due to self intersections). In principle, topology changes can be handled using a more sophisticated tracking algorithm. Second, for exterior flow, if the front becomes very large, the computational cost becomes larger as we require many points to represent the front. We believe that a multi-resolution approach, e.g. using a multi-grid based method could alleviate this problem, but leave further investigation for future work.

3.5.3 Applications.

Visualizing the interior flow with a texture In this experiment we used a texture to visualize the flow in the interior of the domain. We simulated the two phase case, where the boundary of the mesh acts as the interface between the two fluids. After solving for the potentials, we used the potential of the interior domain for moving the interior vertices of the mesh as well as the boundary vertices. After each iteration we resample the boundary and the interior of the mesh and interpolate the texture coordinates. In Fig. 3.14 we show the results for unstable injection.

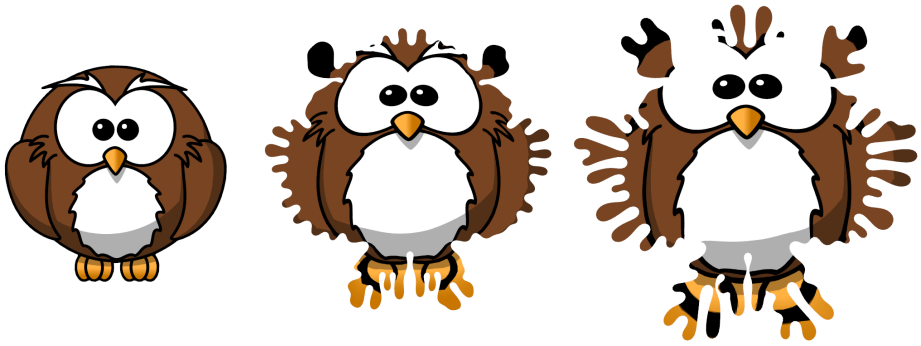


Figure 3.14: Unstable injection from the origin. See text for details.

Pumping from the medial axis. Here we have computed the medial axis of an input curve, and used it as a collection of line singularities from which we pump the fluid (see Fig. 3.15 (left)).

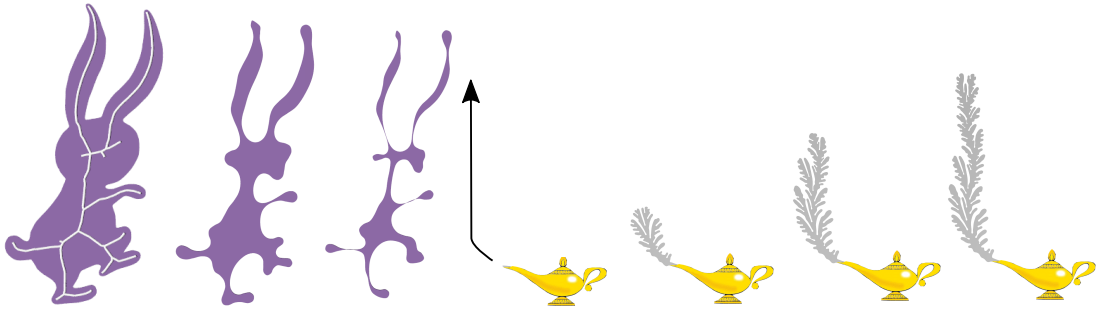


Figure 3.15: (Left) Pumping fluid from the medial axis of the boundary. (Right) Directing the fingers by moving a point singularity.

Controlling the fingers. Here we control the direction which the fingers follow by moving the suction point in exterior flow. In Fig. 3.15 (right) the user moves the suction point in the shown path, and the fingers follow this path as shown in the next images. Note that in the figure we show the caged air in gray and do not show the fluid (which occupies the exterior of the domain).

Obstacles In this experiment we tested exterior flow with multiple obstacles. In Figure 3.16 the fingers are forced to pass through the obstacles as they move toward a line source placed at the bottom. The full simulation is showed in the attached video.

Note that in this simulation we show the air inside the domain in red and do not show the fluid occupying the exterior of the domain.

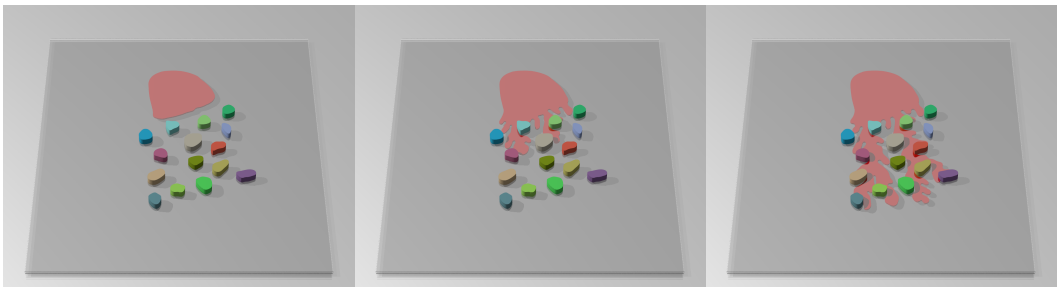
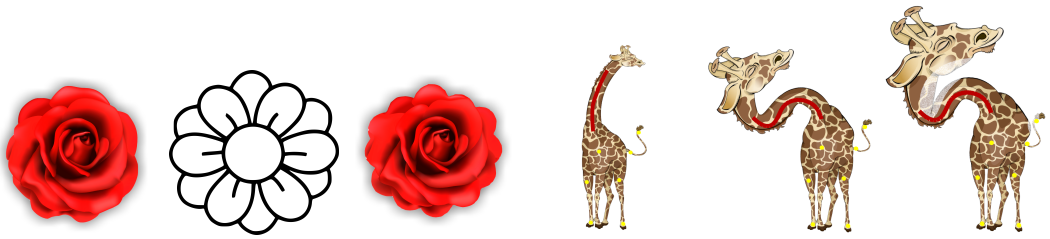


Figure 3.16: Exterior flow with multiple obstacles.

Chapter 4

Iterative Closest Conformal Maps between Planar Domains



(a) Source (b) Target (c) Result (d) Source (e) s2s (f) p2p

Figure 4.1: Left: The rose (a) is given as a source shape and the line drawing (b) as the target. The conformal mapping found via the algorithm is used for transferring the texture from the source shape to the target shape (c). Right: Deformation of the giraffe using stroke-to-stroke constraints, compared to point-to-point constraints. Note that our method (e) generates a deformation with less area distortion (e.g., of the head), and without the foldovers near the head and tail evident when forcing point-to-point matching of points on the curve.

4.1 Background

Conformal maps are often used in computer graphics for mesh parameterization [31] and shape deformation [56] among many other applications. Given two simply connected polygonal domains, the Riemann mapping theorem [46] guarantees that there exists a conformal map between them. Constructing this map efficiently, however, is challenging in practice.

One common approach, is to compose two conformal maps, one from the source to the unit disk, and the second from the unit disk to the target, thus reducing the problem to the case where one of the domains is the unit disk. Different methods were developed for this task, two of the renown ones being the Schwartz-Christoffel method [20] and circle packing [50], where the first yields a continuous map, from any point in the source domain, and the latter requires a discretization of the interior, yet is guaranteed to

converge to the smooth case under refinement [30]. This approach has been further generalized in computer graphics to quasi-conformal [14] maps and bijective harmonic maps [48].

Unfortunately, these methods suffer from a common drawback: as it is necessary to pass through an intermediate convex domain, any distortion incurred in the process could be visible in the final map. Furthermore, the additional degrees of freedom of the problem, namely all Mobius transformations from the unit disk to itself are not considered in this process. Finally, the method should be efficient to allow user control at interactive rates, as well as allow some flexibility since it is not necessarily needed to interpolate the target shape exactly.

We suggest instead to directly map between the source and target domains, by starting with an initial conformal map from the source domain, and iteratively refining it until it matches the boundary of the target domain. We represent a conformal map from the source domain using the *Cauchy transform* [6], which (in the continuous case) spans the entire space of conformal maps from the domain. The Cauchy transform maps continuous functions defined on the boundary of the domain to *holomorphic* functions (complex differentiable functions which are conformal when their derivative does not vanish) in the interior of the domain. Specifically, it can reproduce a holomorphic map from its boundary values. Thus, given the boundary correspondence specified by a conformal map, we can extend this map to the interior of the domain using the Cauchy transform.

We therefore opt for an alternative minimization approach, jointly optimizing for the boundary correspondence and the conformal map. Specifically, we alternate between finding the closest conformal map for a given correspondence and updating the correspondence given a conformal map. This approach leads to a very simple algorithm, which runs at interactive rates and quickly converges to a high quality conformal map. We demonstrate the applicability of our approach using applications to texture transfer and shape deformation. In addition, we show that our method can be easily extended to handle quasi-conformal maps, point-to-point and stroke-to-stroke constraints, and that our results yield lower area distortion than state-of-the-art conformal mapping methods.

4.1.1 Related Work

Many methods for numerically constructing a conformal map from an arbitrary domain to the unit disk can be found in the literature, see e.g. [39] for a recent review. In engineering applications, the most common methods for this task are the Schwarz-Christoffel Mapping [20] and circle packing [50]. However, most of these approaches are quite slow, requiring the solution of non-linear equations or a dense sampling of the domain to achieve sufficient accuracy. Among these, perhaps the closest to our approach is Wegmann's method [58], which also iteratively solves for the boundary correspondence. However, we do not restrict one of the domains to be the unit disk

(which can cause crowding [21]).

In Computer Graphics, planar conformal maps are popular for shape deformation (e.g. [54], [53]), yet existing conformal approaches do not allow for stroke-to-stroke constraints which enable the curve constraints and the boundary to slide as we do. Alternatively, the boundary constraints can be specified by requiring the angles of the input polygon to be preserved [56]. We demonstrate that our approach leads to a better trade-off between the user's constraints and the resulting area distortion. Another iterative approach which does allow boundary sliding is suggested in [22], yet it requires the discretization of the domain and does not output a smooth conformal map.

4.1.2 Contributions

Our main contribution is a fast iterative algorithm for producing conformal maps between two simply connected planar domains, without prescribing boundary correspondence (§4.2). In addition, we:

- Introduce stroke-to-stroke constraints, which can be used for deforming a given shape in an intuitive way and for guiding a conformal map between two domains (§4.3.3).
- Show how to incorporate a quasi-conformal energy to reduce the area distortion (§4.3.2).
- Show applications of our algorithm to image deformation and constrained texture mapping (§4.4).

4.2 Iterative Closest Conformal Mapping

Given an input source shape Ω_s and a target shape Ω_t, both simply connected planar domains, we seek for a conformal map of the source domain, which maps its boundary to the boundary of the target domain. The Riemann mapping theorem [46] states that for any simply connected domain $\Omega \subset \mathbb{C}$, there exists a bijective holomorphic map f from Ω to the unit disk $U = \{z \in \mathbb{C} : |z| < 1\}$. An immediate corollary of the theorem is that for any two simply connected domains Ω_s, Ω_t and the bijections $f_1 : \Omega_s \to U, f_2 : \Omega_t \to U$, one can construct a bijective holomorphic map between the domains $f : \Omega_s \to \Omega_t$, where f is given by $f = f_2^{-1} \circ f_1$.

However, since the space of exact holomorphic maps from one domain to another is quite small and difficult to compute, we would like to relax the problem and gain flexibility to control the behavior of the map. Therefore, we define the following energy, which promotes boundary fitting as a *soft constraint*.

Energy. Given a mapping $f : \Omega_s \to \mathbb{C}$ defined over the source domain Ω_s with boundary curve S, we define an energy for measuring its closeness to the target domain

Ω_t with boundary curve T as:

$$E_c(f) = \oint_S d(f(s), T)^2 ds \tag{4.1}$$

where $d(w, T)$ is the minimal distance from the point w to the boundary of Ω_t: $d(w, T) = \min_{z \in T} |w - z|$.

Discretization. We represent the source and target domains by their boundary, discretized as a source polygon S with n vertices and a target polygon T with m vertices. The space of the holomorphic maps defined over the source polygon is given by a discretization of the *Cauchy transform* for polygons [54], which yields the *Cauchy-Green coordinates*. These coordinates express a subspace of the holomoprhic maps as complex-valued vectors in the range of a fixed complex matrix, depending only on S. Minimizing the discretized energy E_c then boils down to solving a least-squares system using a fixed matrix.

4.2.1 Background - Cauchy-Green coordinates

The Cauchy transform [6] is a widely used operator in complex analysis, which generates the space of holomorphic functions on a domain Ω from continuous functions $f(z)$ defined on the boundary $\partial\Omega$:

$$u(z) = \frac{1}{2\pi i} \oint_{\partial\Omega} \frac{f(w)}{w - z} dw, \quad z \in \Omega. \tag{4.2}$$

The function u is holomorphic in the domain, and when f corresponds to the boundary values of a holomorphic function, the Cauchy transform will reproduce f.

The Cauchy coordinates [54] are a discretized version of the integral (4.2). If we discretize Ω using a polygon $\{z_i\}_{i=1}^n$, then given samples of the function $f_i = f(z_i)$ and by interpolating them linearly on the edges of the polygon, we can calculate the integral analytically. The integration yields n functions $C_i(z)$, denoted as the *Cauchy-Green coordinates*, such that the value of the integral (4.2) is given by:

$$u(z) = \sum_{i=1}^{n} C_i(z) f_i. \tag{4.3}$$

Note, that while the boundary of the domain is discretized as a polygon, the interior of the domain is *continuous*, thus z can take the value of any point in the domain. We provide the expressions for $C_i(z)$ given the input polygon in 2.

4.2.2 Algorithm

Discrete energy. The source polygon is sampled at r points (not necessarily at the vertices), denoted by $\{z_j\}$. If we are given the boundary correspondence to the target, namely for each of the sampled points, we have a corresponding point on the target

domain boundary w_j, then the energy E_c is discretized by:

$$E_c(\{f_i\}) = \sum_{j=1}^{r} |\sum_{i=1}^{n} C_i(z_j) f_i - w_j|^2.$$

Since the correct correspondence is not known in advance, we set w_j as additional optimization variables, and constrain them to be points on the edges of the target polygon T. If we pack the coordinates $C_i(z_j)$ of each sample in a coordinate matrix $C \in \mathbb{C}^{r \times n}$, such that row j contains the coordinates of sample z_j, then the discretized energy is:

$$E_c(\hat{f}, \hat{w}) = \|C\hat{f} - \hat{w}\|^2, \tag{4.4}$$

where $\hat{f} \in \mathbb{C}^n, \hat{w} \in \mathbb{C}^r$ are complex vectors with entries $\{f_i\}, \{w_j\}$, respectively.

Alternating minimization. We obtain the following minimization problem:

$$\min_{\hat{f},\hat{w}} E_c(\hat{f}, \hat{w}), \quad \text{s.t. } \hat{w} \in T. \tag{4.5}$$

We solve the minimization problem using a local-global approach, by alternating between minimizing it with respect to the conformal map given by $C\hat{f}$ (the global step), and the corresponding points on the target, $\hat{w}$ (the local step).

The initial points w^0 are obtained by sampling the target boundary with respect to the arclength of the source boundary sampling, and the coefficients $\hat{f}^0$ are initialized with zero. Then, at each iteration the coefficients $\hat{f}^k$ are updated by minimizing (4.4) with respect to $\hat{f}$. This is a linear least squares problem, where the minimizer is given by $\hat{f}^k = C^+\hat{w}^{k-1}$. Here C^+ is the pseudo-inverse of the matrix C, given by $C^+ = (C^*C)^{-1}C^*$, and C^* is the conjugate transpose of C.

In the second step of each iteration we minimize the energy with respect to $\hat{w}$, under the constraint that $\hat{w}$ is a set of points lying on the target domain boundary. This is done by finding for each point z_j, transformed by the current mapping $\hat{f}^k$, the closest point on an edge (or a vertex) of the target polygon T. The algorithm is summarized in Algorithm 4.1.

Convergence. The Iterative closest conformal map (ICCM) algorithm always converges to a local minimum, since at each iteration the energy is reduced twice. First, by minimizing it with respect to the conformal map given by $\hat{f}$ while keeping $\hat{w}$ fixed and second, by projecting the points on the target polygon T, therefore solving for $\hat{w}$ while keeping $\hat{f}$ fixed. Figure 4.2 shows a typical execution of the algorithm and the energy E_c during the iterations.

Algorithm 4.1 Iterative closest conformal mapping

Input: source polygon S, target polygon T

Output: set of coefficients $\hat{f}$ defining the closest conformal map found from S to T

$\hat{z} = \text{SamplePolygon(S)}$
$\hat{w}^0 = \text{SamplePolygon(T)}$
$\hat{f}^0 = 0$
$k \leftarrow 0$
while $E_c(\hat{f}^k, \hat{w}^k) > threshold$ **do**
    $\hat{f}^{k+1} \leftarrow C^+ \hat{w}^k$
    $\hat{w}^{k+1} \leftarrow \text{ClosestPoint}(T, C\hat{f}^{k+1})$
    $k \leftarrow k + 1$
end while
$\hat{f} \leftarrow \hat{f}^k$

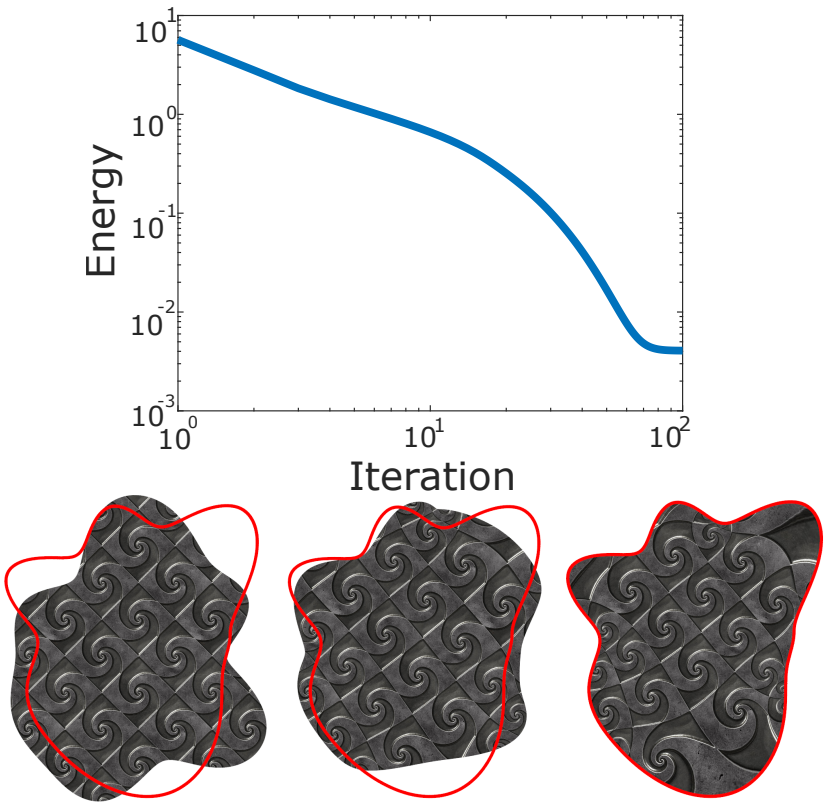


Figure 4.2: Top: the energy during the iterations of a typical execution of the algorithm on a log-log scale. Bottom, from left to right: the initial domain, the conformal mapping after a single iteration of the algorithm, and the conformal mapping after 100 iterations.

4.3 Extensions

4.3.1 P2P Constraints

For gaining more control over the behavior of the conformal map, we can add additional energies to the minimization problem. One possibility is the point-to-point energy E_{P2P} which allows the user to guide the conformal map by specifying points in the source

domain u_i and their desired location in the target domain v_i. Packing the coordinates of the source points u_i together, we obtain the matrix C_{P2P} where each row contains the coordinates of a source point. Then, the P2P energy is expressed by:

$$E_{P2P}(\hat{f}) = \|C_{P2P}\hat{f} - \hat{v}\|^2 \tag{4.6}$$

where $\hat{v}$ is the vector of target locations. Minimizing only this energy yields a conformal map which transforms the chosen source points as close as possible to the target positions. With both energies combined, we now seek to minimize $E = E_c(\hat{f}, \hat{w}) + \lambda E_{P2P}(\hat{f})$, where λ is a parameter controlling the strength of the P2P energy. This optimization problem can be solved in a very similar way to the ICCM algorithm, the only difference being the fact that now the minimization with respect to $\hat{f}$ should be taken over the weighted least squares problem defined by the two energies. In addition, for the initialization step it is beneficial to calculate $\hat{f}^0$ by minimizing the E_{P2P} energy, yielding a rough initial map, and set $\hat{w}^0$ to be the closest points on the target boundary to the sampled points mapped by $\hat{f}^0$. Figure 4.3 shows the result of running the algorithm with user defined P2P constraints.

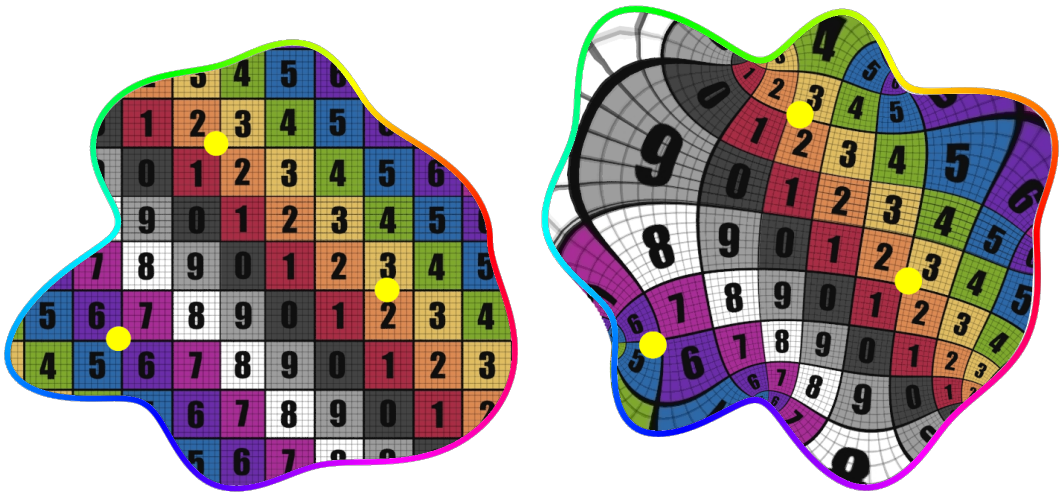


Figure 4.3: left: source domain and constrained points, right: the mapping obtained by the ICCM algorithm with the specified target boundary and P2P constraints.

4.3.2 Quasi-conformal maps

Since the number of degrees of freedom for a conformal map from one simply-connected domain to another is quite small, a possible way to extend the space of mappings is allowing the map to be quasi-conformal. Instead of looking for a holomorphic function, we will search for a *complex harmonic* function, where the real and imaginary parts are both real harmonic functions. It is known that any complex harmonic function f can be decomposed as the sum of holomorphic and antiholomorphic functions. Thus, we will

represent the complex harmonic function f using the Cauchy-Green coordinates by:

$$f(z) = \Phi(z) + \Psi(z) = \sum_{j=1}^{n} C_j(z)\phi_j + \sum_{j=1}^{n} \overline{C_j(z)}\overline{\psi_j} \tag{4.7}$$

where $\Phi(z) = \sum_{j=1}^{n} C_j(z)\phi_j$ is a holomorphic function and $\Psi(z) = \sum_{j=1}^{n} \overline{C_j(z)}\overline{\psi_j}$ is an antiholomorphic function. Denoting by C the matrix of coordinates for the sampled points (similarly to the previous section), the energy E_c now becomes:

$$E_c = \left\| \begin{pmatrix} C & \overline{C} \end{pmatrix} \begin{pmatrix} \hat{\phi} \\ \hat{\psi} \end{pmatrix} - \hat{w} \right\|^2 \tag{4.8}$$

where $\hat{\phi}$ is the vector with entries ϕ_j and $\hat{\psi}$ is the vector with entries $\overline{\psi_j}$.

The dilatation of a mapping f is defined as [2]:

$$D_f(z) = \frac{|f_z| + |f_{\overline{z}}|}{|f_z| - |f_{\overline{z}}|} = \frac{1 + {|f_{\overline{z}}|}/{|f_z|}}{1 - {|f_{\overline{z}}|}/{|f_z|}} \tag{4.9}$$

where $f_z = (\frac{\partial}{\partial x} + i\frac{\partial}{\partial y})f$ and $f_{\overline{z}} = (\frac{\partial}{\partial x} - i\frac{\partial}{\partial y})f$. The dilatation can be used for measuring the conformal distortion of the mapping: for a conformal map the dilatation is exactly 1 (since $f_{\overline{z}} = 0$, which is equivalent to satisfying the Cauchy-Riemann equations), and as it gets larger, the map distorts angles more. Therefore, for limiting the amount of conformal distortion the dilatation has to be minimized, which can be achieved by minimizing $|f_{\overline{z}}|$. Note that while a quasi-conformal map requires a bounded dilatation in the whole domain, we do not find this global bound, but attempt to minimize it by minimizing the values of $|f_{\overline{z}}|$ on the boundary. From the maximum modulus principle [38], since in our case $f_{\overline{z}}$ is antiholomorphic, the maximum value of $|f_{\overline{z}}|$ occurs on the boundary of the domain, and thus by minimizing $|f_{\overline{z}}|$ on the boundary, it will be minimized inside the domain as well.

In our representation, since f decomposes as a sum of two holomorphic and antiholomorphic functions we get that $f_z = \Phi_z = D\hat{\phi}$ and $f_{\overline{z}} = \Psi_{\overline{z}} = \overline{D}\hat{\psi}$, where D is the matrix with the derivatives of the coordinates for the sampled points, i.e. $D = \frac{\partial}{\partial z}C$. Therefore, we add the energy:

$$E_q = \|f_{\overline{z}}\|^2 = \|\overline{D}\hat{\psi}\|^2, \tag{4.10}$$

where the derivative is calculated for each element in the matrix, as described in 2. Figure 4.4 compares the area and conformal distortion of the conformal and quasi-conformal maps achieved using our approach. Note the reduced area distortion near the boundary of the domain, at the expense of a small conformal distortion.

4.3.3 Stroke to Stroke mapping

A similar idea to ICCM can be employed for extending the point-to-point constraints to curve-to-curve constraints. In this type of constraints, the user can draw a source

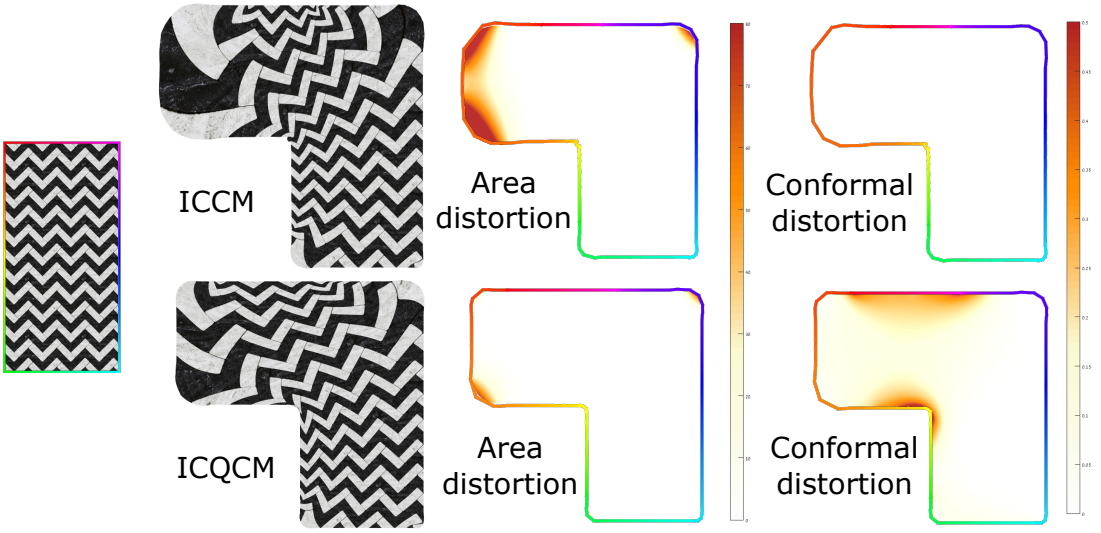


Figure 4.4: Left: The source polygon. Top row: The conformal map found by the ICCM algorithm, bottom row: quasi-conformal map found by the ICQCM algorithm. Note that we achieve less area distortion in the quasi-conformal mapping at the expense of some conformal distortion.

stroke Sk_s inside the domain and a target stroke Sk_t, and the goal would be to find a conformal mapping of the domain which maps between the drawn strokes. Formally, we define the energy E_{S2S} in a very similar way to the energy E_c:

$$E_{S2S}(f) = \int_{Sk_s} d(f(s), Sk_t)^2 ds, \tag{4.11}$$

where $d(z, T)$ measures the minimal distance of the point z from the curve T. The discretization of the energy is achieved using the Cauchy-Green representation of the conformal mapping f. First, the source and the target strokes are sampled uniformly at n points $\{s_j\}_{j=1}^n$ and $\{t_j\}_{j=1}^n$. Next, we calculate the Cauchy-Green coordinates for the points sampled on the source stroke and pack them together in a matrix C_{sk}. Finally, the discretized energy is defined by:

$$E_{S2S}(\hat{f}, \hat{t}) = \|C_{sk}\hat{f} - \hat{t}\|^2 \tag{4.12}$$

where $\hat{t}$ is the vector of points sampled on the target stroke. The minimization of this energy is done using a similar iterative algorithm to ICCM, where at each iteration a least squares problem is solved for the coefficients $\hat{f}^{k+1}$, and then the new set of points $\hat{t}^{k+1}$ is calculated by projecting the current mapping of the source points on the target stroke.

This energy, similarly to the P2P energy, can be used for guiding the conformal map when combined with the closeness energy E_c, but can also be useful for deforming a shape. In the latter scenario, it is beneficial to add a regularization term which is defined as an additional energy $E_s(\hat{f}) = \|D^{(2)}\hat{f}\|^2$, where $D^{(2)}$ is a matrix containing

(a) Source (b) p2p (c) s2s

Figure 4.5: Stroke to stroke constraints used for shape deformation. Left: the original shape and the given constraints. Middle: deformation with point-to-point constraints (the points on the strokes are fixed). Right: deformation with stroke-to-stroke constraints (points are allowed to move along the strokes). Note that using the stroke-to-stroke constraints yields a lower area distortion for the hands and the head.

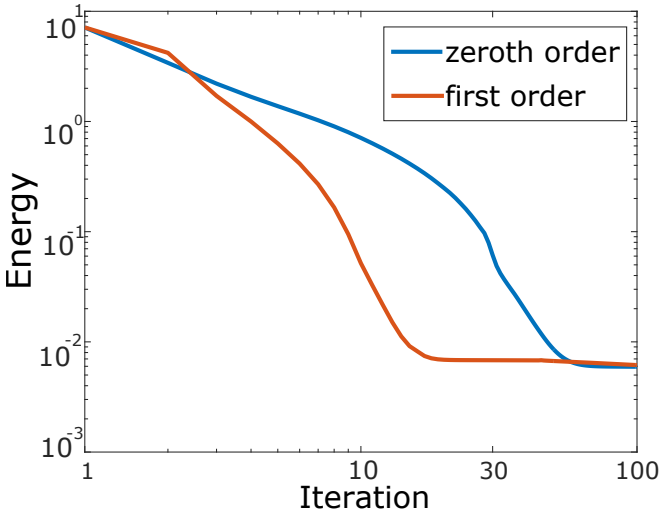
the second derivative of the Cauchy-Green coordinates for additional points sampled on the boundary of the domain. This energy is useful for fixing the degrees of freedom (when the number of constraints is smaller than the number of coordinate functions) and for preserving the smoothness of the boundary. Figure 4.5 shows the deformation found using the stroke-to-stroke constraints, and the comparison to the deformation found using similar point-to-point constraints. Note that since the points are allowed to move freely on the target stroke, the deformation found using the stroke-to-stroke constraints yields a smaller area distortion for the hands and the head.

4.3.4 Higher Order Approximation of the Distance Function

In the global step of the minimization problem (the optimization for $\hat{f}$), we used a zeroth order approximation of the distance function $d(z,T)$ at the mapped point $f(z_j)$, namely the squared distance to the closest point found in the previous step w_j. One could use, alternatively, a higher order approximation for the distance function as suggested in [44]. The first order approximation of $d(z,T)$ is the distance to the tangent at the closest point w_j. Thus, using complex-variable notation the first order approximation is given by $d(z,T) \approx \mathrm{Re}\left((z-w_j)\overline{N_j}\right)^2$, where N_j is the unit normal at w_j and we have used the representation of a dot product between two vectors $a,b \in \mathbb{R}^2$ in complex form: $\langle a,b\rangle = \mathrm{Re}(a\bar{b})$. Integrating the first order approximation in the global step, we obtain the following minimization problem:

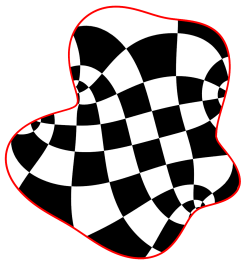
$$\hat{f}^{k+1} = \underset{\hat{f}}{\mathrm{argmin}} \| \mathrm{Re}\left(\overline{N}\left(C\hat{f}-\hat{w}\right)\right)\|^2 + \lambda\|C\hat{f}-\hat{w}\|^2 \tag{4.13}$$

where N is the diagonal matrix with the entries N_j on its main diagonal. Note that the zeroth order approximation is used here as a regularization term for stabilizing the energy when the algorithm is close to converge. This is still a simple least squares problem which can be solved by converting the complex variable formula to one with real variables. Note that the local step does not need to change since the distance function can be exactly calculated when the points $f(z_j)$ are fixed, therefore an approximation is not necessary. Figure 4.6 shows a comparison between the different approximation orders. Notice that while the zeroth order approximation works well for points far away from the curve, the first order approximation behaves better for points close to the curve, and the convergence is achieved much faster.



(a) Convergence of the energy E_c

(b) zeroth order



(c) first order

Figure 4.6: (a) Convergence of the energy using different orders of approximation for the distance function. (b), (c) the conformal map obtained by the zeroth and first order approximations at iteration 30, in which the first order approximation has converged.

4.4 Experimental Results

4.4.1 Implementation Details

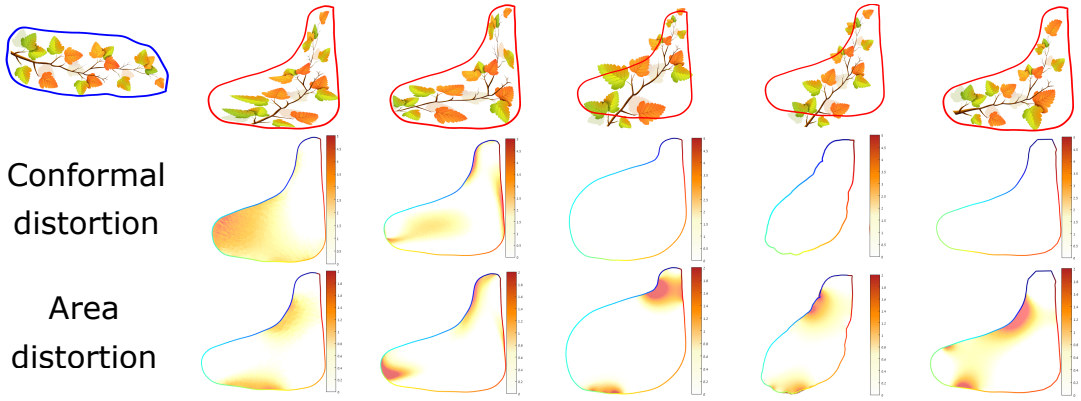
We have implemented the ICCM algorithm and its extensions in MATLAB. Given a source polygon S with n vertices and a target polygon T, we sample the source polygon at r points for creating the vector of points P used for discretizing the closeness energy E_c. It is necessary to sample points at the edges of the polygon for constraining their mapping to be close to the target polygon. While a sparse sampling may result in mapping of points between the sampled ones to points far from the target polygon, sampling too dense might make the algorithm slow. The number of sampled points per edge should depend on its length and on the available computational resources. Since

in our experiments the source polygons had approximately uniform edge lengths, we have found that sampling each edge at 4 points (including the vertices) is sufficient for achieving good results. The target polygon is sampled according to the accumulative arclength of the points in P, starting from the first point P_1. Next, we calculate the coordinates matrix C of size $r \times n$, where row j contains the coordinates of point P_j. The other energies E_s, E_{P2P}, E_{S2S} are discretized in a similar way, with the coordinate matrices $D_s^{(2)}, C_{P2P}, C_{S2S}$. Finally, the conformal map is found by minimizing the combined energy:

$$\begin{aligned} E(\hat{f}) &= E_c(\hat{f}, \hat{w}) + \alpha E_s(\hat{f}) + \beta E_{P2P}(\hat{f}) + \gamma E_{S2S}(\hat{f}, \hat{t}) \\ &= \|C\hat{f} - \hat{w}\|^2 + \alpha\|D^{(2)}\hat{f}\|^2 + \beta\|C_{P2P}\hat{f} - \hat{v}\|^2 + \gamma\|C_{S2S}\hat{f} - \hat{t}\|^2 \end{aligned}$$

Note that $E(\hat{f})$ is a quadratic energy in $\hat{f}$, and thus can be written as $E(\hat{f}) = \|A\hat{f} - b\|^2$. Since the coordinates matrices are constant during the iterations of the algorithm, we can calculate the pseudo-inverse of A in advance, and use it during the iterations minimizing the energy with respect to $\hat{f}$ by multiplying $\hat{f}^{k+1} = A^+ b^k$. After the new coefficients $\hat{f}^{k+1}$ are found at each iteration, the vector b^{k+1} is calculated by updating $\hat{w}^{k+1}$ and $\hat{t}^{k+1}$ to be the closest points on the target boundary and target strokes.

In the case of quasi-conformal mapping, the vector of coefficients $\hat{f}$ (with n elements in the conformal case), is extended to contain $2n$ elements $\hat{f}^q = \begin{pmatrix} \hat{\phi} \\ \hat{\psi} \end{pmatrix}$, where the first n elements represent the holomorphic function $\phi(z)$ and the last n elements represent the antiholomorphic function $\psi(z)$. Additionally, each one of the coordinate matrices is concatenated from the right with its conjugate (i.e. $C^q = \begin{pmatrix} C & \overline{C} \end{pmatrix}$), so that the quasi-conformal function evaluated at the sampled points is given by $f(\hat{z}) = C^q \hat{f}^q$. In this case, the energy E_q is also added as part of the weighted least squares problem.



(a) Source (b) [48] (c) [22] (d) [56] (e) Szego [54] (f) ICCM

Figure 4.7: Comparison of our algorithm to state-of-the-art methods for computing maps between planar domains. Note that our algorithm produces a conformal map with no shearing artifacts, but may introduce some area distortions.

4.4.2 Limitations

One limitation of our method is that it does not have the option to control the area distortion. Therefore, the mappings found by the method may introduce large area distortions in order to minimize the energy. However, by searching for a quasi conformal mapping we have seen that the area distortion can be reduced at the expense of some conformal distortion.

In addition, our method does not prevent flipping, which can appear in the continuous holomorphic map obtained by the algorithm, and may not preserve the order of the points on the boundary. Furthermore, the algorithm depends on the initial boundary correspondence, and for a bad initialization, e.g., a convex part of the source is mapped into two different parts of the target, the optimization will converge to a local minimum.

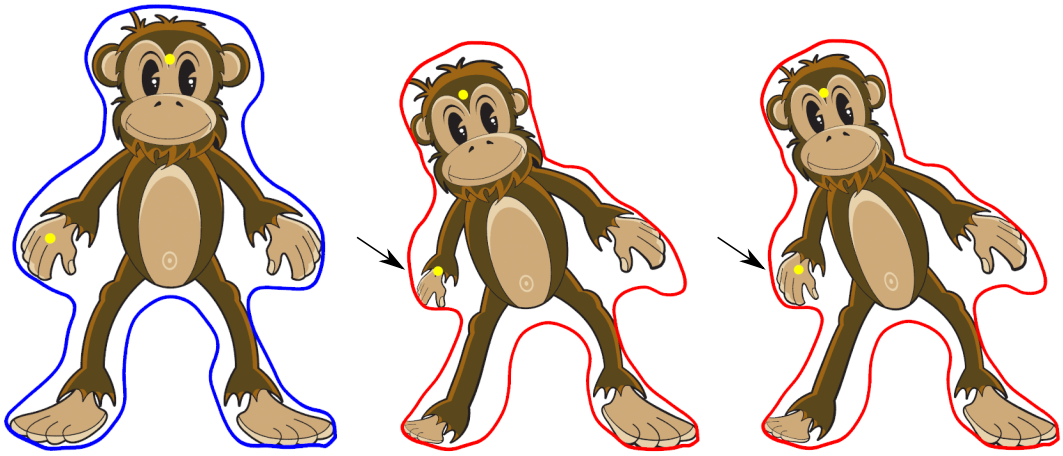
4.4.3 Comparisons

We have compared our algorithm to several methods for mapping between planar shapes. Figure 4.7 shows the source polygon in blue, the target polygon in red, the mapping achieved with each of the methods, and the conformal and area distortions. In column (b) we show the results of the method described in [48] for constructing a smooth bijective map between arbitrary polygons via a convex regular polygon. We have discretized the interior of the two domains, constructed two harmonic maps from the domains into a convex regular polygon, and composed one with the inverse of the other in order to get a mapping between the two domains. In column (c) we show the results of the method in [22], in which the interior of the domain is discretized, and an energy which measures the distance between the mapping of the source's boundary to the target's boundary and the distortion of the laplacian of the source mesh is minimized. Notice that both of these methods do not produce a conformal map and therefore introduce some conformal distortion. In column (d), we use the method from [56] for finding a conformal mapping which maps the angles of the source polygon to the angles of the target polygon. However, the length of the edges is not prescribed, and therefore the mapping does not interpolate the target polygon and can produce large area distortion as can be seen in the results. In column (e) we have used the Szego coordinates which were introduced in [54]. Our method is shown in column (f). In the example we have also used two p2p constraints in order to construct the initial boundaries correspondence and for guiding the source of the branch and the top leaf to their desired location. Notice that the method produces a conformal map, therefore there is no conformal distortion, but the area distortion is not constrained and therefore it introduces more area distortion than the other methods in some parts of the mesh.

4.4.4 Additional Results

Deformations. Figure 4.8 shows a deformation generated by mapping a source sketch to a target sketch, both drawn by the user. Two point-to-point constraints were used

for guiding the mapping of the hand and the head of the monkey. Note that the quasi-conformal map better approximates the point-to-point constraints, as well as reduces the area distortion (see the point constraint in the hand).



(a) Source (b) Conformal (c) Quasi-conformal

Figure 4.8: Deformation of the monkey using conformal vs quasi-conformal mappings. Note that the quasi-conformal map better approximates the point-to-point constraints, as well as reduces the area distortion (see the point constraint in the hand).

Constrained texture mapping. Figure 4.9 shows how the algorithm can be used for constrained texture mapping. In this experiment we have used the method from [7] for calculating a conformal flattening of the given mesh, resulting in a 2D mesh which is conformal to the original. Next, we used our algorithm for finding a conformal map from the boundary of the texture (a square) to the boundary of the 2D mesh, and the point-to-point constraints were used for guiding the conformal map.

Texture transfer. In this experiment we calculated a conformal mapping from a hexagon into a flattened mesh, which is conformal to the target 3D mesh. Then, we have transfered the texture from the original shape to the target shape through composition of the two conformal maps. The results are shown in Figure 4.10.

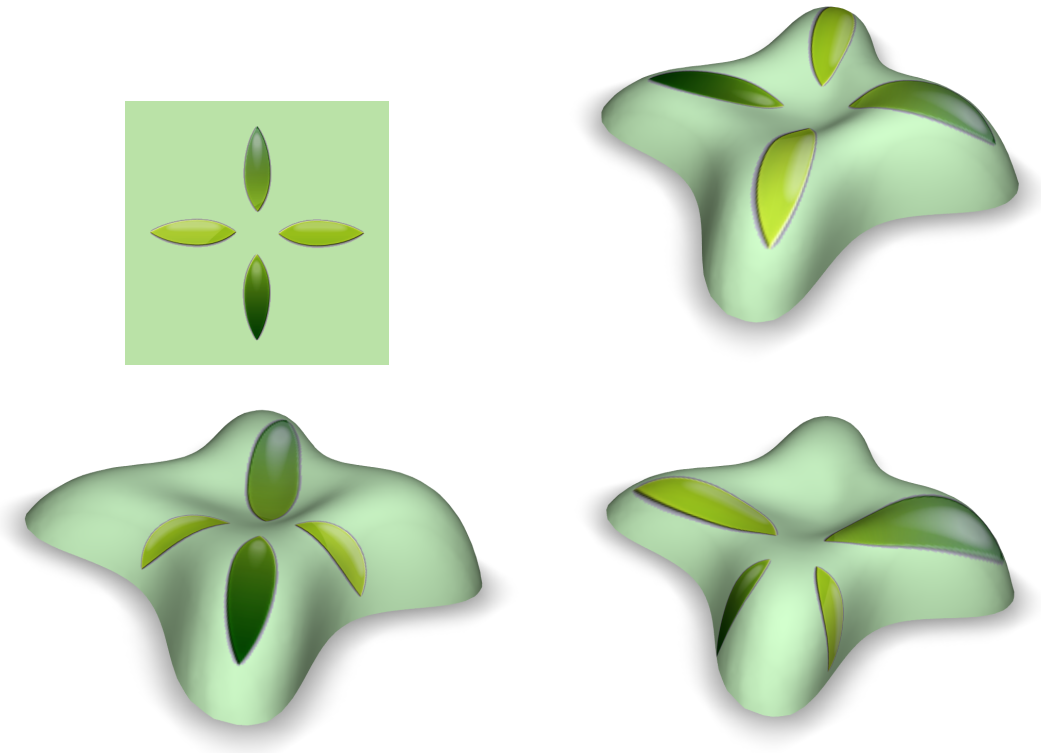
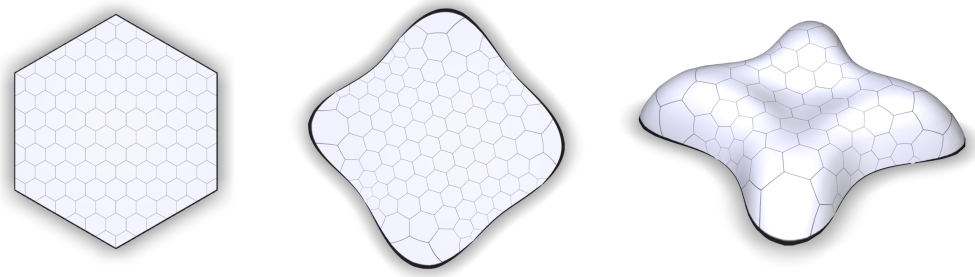


Figure 4.9: Constrained texture mapping. Top left: the input texture. Top right and bottom row: the texture is mapped to the surface, using point-to-point constraints as a guidance.



(a) Source (b) Flattened target (c) Target mesh

Figure 4.10: Texture transfer between shapes. The texture from the hexagon (a) was transfered to the lilium mesh (c) through composition of two conformal maps.

Chapter 5

Conclusion and open questions

5.1 Conclusion

In this work, the Cauchy-Green coordinates were used for efficiently solve 2D problems which can be expressed in a boundary integral formulation. The fast algorithm allows adding user interaction in real time, allowing the user to guide the result of the algorithm toward the desired solution. Specifically, we showed how the Hele-Shaw flow can be formulated in a boundary integral formulation which the Cauchy-Green coordinates can be integrated into, yielding an efficient discrete scheme for simulating the flow. The user can choose between different types of singularities and move them in real time in order to control the flow while preserving the physically correct behavior. Additionally, we showed how the Cauchy-Green coordinates are applicable to the exterior of a planar bounded domain and to a multiply connected domain, which allows us to simulate the two phase flow and obstacles.

The second application we showed for the Cauchy-Green coordinates is conformal mapping between planar domains. In this problem, we used the coordinates for parameterizing a space of conformal maps, and devise an iterative algorithm for finding the conformal map from this space which maps the source domain the closest to the target domain. The efficiency of this algorithm allows the user to interactively modify constraints such as point-to-point and stroke-to-stroke correspondences, and thus guide the algorithm toward the desired mapping. Furthermore, we showed how this algorithm can be generalized to quasi-conformal maps, for enriching the space of mappings and reducing the amount of area distortion.

5.2 Open questions

Extension to 3D The Cauchy-Green coordinates derive from the Cauchy integral formula in complex analysis and thus are limited to two dimensions. However, there is an extension of Cauchy integral formula to higher dimensions [18] and in particular, there is a discretization in 3D [8]. It may be interesting to use these coordinates for the applications described in this work for simulating 3D flows and finding mappings

between surfaces.

Different types of flows The Hele-Shaw flow could be written in a boundary integral formulation since it involves a harmonic function which is the solution of Laplace's equation which is conformally invariant (required in the interface tracking method) and can be represented as the real part of a holomorphic function (required in the potential representation method). However, there are other types of equations which are also conformally invariant [5] and there are other types of complex barycentric coordinates which are not necessarily holomorphic [55]. Therefore, it might be possible to extend this work to different types of flows by expressing them in a boundary integral formulation using other type of barycentric coordinates.

Preserving points order to prevent flips In 4 we described an efficient algorithm for calculating conformal maps between planar domains. The algorithm was shown to converge, however it might produce flips and double-covers of the target domain, and the boundary correspondence may not preserve the order of the points on the boundary. Therefore, it might be interesting to investigate the behavior of the algorithm when the preservation of the points order is added as a hard constraint, allowing the points to slide on the target boundary without crossing each other. Adding this as a hard constraint may prevent flips and help the algorithm converge into a correct mapping.

Bibliography

[1] Lars V Ahlfors. *Complex analysis*. 1966.

[2] Lars Valerian Ahlfors and Clifford J Earle. Lectures on quasiconformal mappings. 1966.

[3] Omri Azencot, Orestis Vantzos, Max Wardetzky, Martin Rumpf, and Mirela Ben-Chen. Functional thin films on surfaces. In *Proc. of SCA*, pages 137–146. ACM, 2015.

[4] Christopher Batty, Andres Uribe, Basile Audoly, and Eitan Grinspun. Discrete viscous sheets. 31(4):113, 2012.

[5] Martin Z Bazant. Conformal mapping of some non-harmonic functions in transport theory. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 460, pages 1433–1452. The Royal Society, 2004.

[6] Steven R Bell. *The Cauchy transform, potential theory and conformal mapping*. CRC press, 2015.

[7] Mirela Ben-Chen, Craig Gotsman, and Guy Bunin. Conformal flattening by curvature prescription and metric scaling. In *Computer Graphics Forum*, volume 27, pages 449–458. Wiley Online Library, 2008.

[8] Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. Variational harmonic maps for space deformation. In *ACM Transactions on Graphics (TOG)*, volume 28, page 34. ACM, 2009.

[9] David Bensimon, Leo P Kadanoff, Shoudan Liang, Boris I Shraiman, and Chao Tang. Viscous flows in two dimensions. 58(4):977, 1986.

[10] Miklós Bergou, Basile Audoly, Etienne Vouga, Max Wardetzky, and Eitan Grinspun. Discrete viscous threads. 29(4):116, 2010.

[11] Irmgard Bischofberger, Radha Ramachandran, and Sidney R Nagel. An island of stability in a sea of fingers: emergent global features of the viscous-flow instability. 11(37):7428–7432, 2015.

[12] Robert Bridson. *Fluid simulation for computer graphics*. CRC Press, 2015.

[13] Tyson Brochu, Todd Keeler, and Robert Bridson. Linear-time smoke animation with vortex sheet meshes. In *Proc. of SCA*, pages 87–95. Eurographics Association, 2012.

[14] Renjie Chen and Ofir Weber. Bounded distortion harmonic mappings in the plane. *ACM Transactions on Graphics (TOG)*, 34(4):73, 2015.

[15] Linda Cummings, Stefan Llewellyn Smith, Paul Martin, and Bartosz Protas. Modern applications of complex variables: Modeling, theory and computation. 2015.

[16] Michael C Dallaston. Mathematical models of bubble evolution in a hele-shaw cell. 2013.

[17] Michael Charles Dallaston and Scott McCue. An accurate numerical scheme for the contraction of a bubble in a hele–shaw cell. 54:309–326, 2013.

[18] AJ Davies, R Foot, GC Joshi, and BHJ McKellar. Quaternionic methods in integral transforms of geophysical interest. *Geophysical Journal International*, 99(3):579–582, 1989.

[19] Brett Desbenoit, Eric Galin, and Samir Akkouche. Simulating and modeling lichen growth. In *Computer Graphics Forum*, volume 23, pages 341–350. Wiley Online Library, 2004.

[20] Tobin A Driscoll and Lloyd N Trefethen. *Schwarz-Christoffel Mapping*, volume 8. Cambridge University Press, 2002.

[21] Tobin A Driscoll and Stephen A Vavasis. Numerical conformal mapping using cross-ratios and delaunay triangulation. *SIAM Journal on Scientific Computing*, 19(6):1783–1803, 1998.

[22] Mathias Eitz, Olga Sorkine, and Marc Alexa. Sketch based image deformation. In *VMV*, pages 135–142. Citeseer, 2007.

[23] Michael S Floater. Generalized barycentric coordinates and applications. 24:161–214, 2015.

[24] LA Galin. Unsteady filtration with a free surface. In *Dokl. Akad. Nauk USSR*, volume 47, pages 246–249, 1945.

[25] Abhinav Golas, Rahul Narain, Jason Sewall, Pavel Krajcevski, Pradeep Dubey, and Ming Lin. Large-scale fluid simulation using velocity-vorticity domain decomposition. 31(6):148, 2012.

[26] Björn Gustafsson. On a differential equation arising in a hele shaw flow moving boundary problem. 22(1):251–268, 1984.

[27] Bjorn Gustafsson and Alexander Vasilev. *Conformal and potential analysis in Hele-Shaw cells*. Springer Science & Business Media, 2006.

[28] David Hahn and Chris Wojtan. High-resolution brittle fracture simulation with boundary elements. 34(4):151, 2015.

[29] Antony Hall. Hele-shaw cell problem 2013. `http://www.antonyhall.net/works01/heleshaw02.html`, 2013.

[30] Zheng-Xu He and Oded Schramm. On the convergence of circle packings to the riemann map. *Inventiones mathematicae*, 125(2):285–305, 1996.

[31] Kai Hormann, Bruno Lévy, and Alla Sheffer. Mesh parameterization: Theory and practice. 2007.

[32] Sam D Howison. A note on the two-phase hele-shaw problem. 409:243–249, 2000.

[33] HS Kasana. *Complex Variables: Theory and Applications*. PHI Learning Pvt. Ltd., 2005.

[34] Todd Keeler and Robert Bridson. Ocean Waves Animation using Boundary Integral Equations and Explicit Mesh Tracking. In Vladlen Koltun and Eftychios Sifakis, editors, *Proc. of SCA*, 2014.

[35] AH Khalid. *Free boundary problems in a Hele-Shaw cell*. PhD thesis, UCL (University College London), 2015.

[36] Theodore Kim, Jason Sewall, Avneesh Sud, and Ming C Lin. Fast simulation of laplacian growth. 27(2):68–76, 2007.

[37] Theodore Won-Hyung Kim. *Physically-based simulation of ice formation*. PhD thesis, University of North Carolina at Chapel Hill, 2006.

[38] Steven G Krantz. *Handbook of complex variables*. Springer Science & Business Media, 2012.

[39] Prem Kythe. *Computational conformal mapping*. Springer Science & Business Media, 2012.

[40] Shuwang Li, John S Lowengrub, Jake Fontana, and Peter Palffy-Muhoray. Control of viscous fingering patterns in a radial hele-shaw cell. 102(17):174501, 2009.

[41] Shuwang Li, John S Lowengrub, and Perry H Leo. A rescaling scheme with application to the long-time simulation of viscous fingering in a hele–shaw cell. 225(1):554–567, 2007.

[42] Yaron Lipman, David Levin, and Daniel Cohen-Or. Green coordinates. In *ACM Trans. on Graphics (TOG)*, volume 27, page 78. ACM, 2008.

[43] NervousSystem. Laplacian growth 2d. `http://n-e-r-v-o-u-s.com/projects/albums/laplacian-growth-2d/`, 2012.

[44] Helmut Pottmann, Stefan Leopoldseder, and Michael Hofer. Registration without icp. *Computer Vision and Image Understanding*, 95(1):54–71, 2004.

[45] Linhai Qiu, Yue Yu, and Ronald Fedkiw. On thin gaps between rigid bodies two-way coupled to incompressible flow. *Journal of Computational Physics*, 292:1–29, 2015.

[46] Walter Rudin. *Real and complex analysis*. Tata McGraw-Hill Education, 1987.

[47] PG Saffman. Viscous fingering in hele-shaw cells. 173:73–94, 1986.

[48] Teseo Schneider and Kai Hormann. Smooth bijective maps between arbitrary planar polygons. *Computer Aided Geometric Design*, 35:243–254, 2015.

[49] Aviv Segall and Mirela Ben-Chen. Iterative Closest Conformal Maps between Planar Domains. *Computer Graphics Forum*, 2016.

[50] Kenneth Stephenson. The approximation of conformal structures via circle packing. *SERIES IN APPROXIMATIONS AND DECOMPOSITIONS*, 11:551–582, 1999.

[51] Robert Walker Sumner. Pattern formation in lichen, 2001.

[52] Tetsuya Takahashi, Yoshinori Dobashi, Issei Fujishiro, Tomoyuki Nishita, and Ming C Lin. Implicit formulation for sph-based viscous fluids. In *Computer Graphics Forum*, volume 34, pages 493–502. Wiley Online Library, 2015.

[53] Amir Vaxman, Christian Müller, and Ofir Weber. Conformal mesh deformations with möbius transformations. *ACM Transactions on Graphics (TOG)*, 34(4):55, 2015.

[54] Ofir Weber, Mirela Ben-Chen, and Craig Gotsman. Complex barycentric coordinates with applications to planar shape deformation. In *Computer Graphics Forum*, volume 28, pages 587–597. Wiley Online Library, 2009.

[55] Ofir Weber, Mirela Ben-Chen, Craig Gotsman, and Kai Hormann. A complex view of barycentric mappings. In *Computer Graphics Forum*, volume 30, pages 1533–1542. Wiley Online Library, 2011.

[56] Ofir Weber and Craig Gotsman. Controllable conformal maps for shape deformation and interpolation. In *ACM Transactions on Graphics (TOG)*, volume 29, page 78. ACM, 2010.

[57] Ofir Weber and Denis Zorin. Locally injective parametrization with arbitrary fixed boundaries. *ACM Transactions on Graphics (TOG)*, 33(4):75, 2014.

[58] Rudolf Wegmann. An iterative method for conformal mapping. *Journal of computational and applied mathematics*, 14(1):7–18, 1986.

[59] Changxi Zheng and Doug L James. Harmonic fluids. 28(3):37, 2009.

[60] Yufeng Zhu, Robert Bridson, and Chen Greif. Simulating rigid body fracture with surface meshes. 34(4):150, 2015.

ניתן לסמלץ מודל של בועת אוויר הכלואה בתוך נוזל יותר צמיג, ובאמצעות שילוב של ייצוג זה עם הייצוג בתוך תחום סגור ניתן לסמלץ את זרימת הילי־שו עם שני נוזלים. הייצוג של הקואורדינטות בתחומים שאינם פשוטי קשר מאפשר לסמלץ זרימת הילי־שו עם מכשולים, כלומר, אזורים שהנוזלים לא יכולים לחדור לתוכם. הרחבות אלו מאפשרות סימולציה של סוגים שונים של זרימות הניתנות לשליטה על ידי המשתמש וכתוצאה מכך יצירה של מגוון רחב של תבניות שונות.

הבעיה השניה שנעסוק בה בעבודה זו היא הבעיה של מציאת מיפויים בין שני תחומים דו מימדיים. בין היישומים של פתרון בעיה זו ניתן למצוא העברה של טקסטורה מתחום אחד לשני, דפורמציה של תמונה על פי סקיצה המצויירת על ידי המשתמש ועיוותים של תמונות. המיפוי שנמצא בין התחומים בדרך כלל נדרש לקיים מספר תכונות, כאשר בין החשובות מביניהן הן שמירה על הזוויות בין שני וקטורים היוצאים מאותה נקודה (באופן לא פורמלי, תכונה זו ניתנת לתיאור כשמירה על הנאמנות למקור של הטקסטורה) ושהעיוות של השטח שנוצר מהמיפוי יהיה קטן. הכמות של עיוות הזווית בין שני וקטורים נקרא העיוות הקונפורמי של המיפוי, ומיפוי קונפורמי הוא מיפוי שאינו מעוות את הזוויות באף נקודה. הגישות הקודמות לפתירת הבעיה בדרך כלל כללו קיבוע של ההתאמה בין השפה של תחום המקור והשפה של תחום היעד, ותהליך המנסה למזער את העיוות הקונפורמי ועיוות השטח הנוצר בתוך התחום. באחת מהעבודות הקודמות ההתאמה בין השפות של התחומים השתנתה במהלך האלגוריתם כדי למזער את העיוות קונפורמי, אבל היא כללה (בדומה לרוב העבודות הקודמות) דיסקרטיזציה של התחום כולו ולכן התוצאה של האלגוריתם היתה תלויה ברזולוציה של הדגימה של התחום והמיפוי שהתקבל לא היה קונפורמי. מספר שיטות אחרות מחפשות עבור מיפוי קונפורמי אל מעגל היחידה, כמו דחיסת מעגלים ומיפוי שוורץ־כריסטופל. בכל אופן, שיטת הדחיסה של המעגלים כוללת אלגוריתם איטרטיבי המתכנס מאוד לאט ומציאת מיפוי שוורץ־כריסטופל כוללת פתירה של מערך משוואות לא לינארית, תהליך שנהיה איטי עבור פוליגונים גדולים.

השיטה שלנו, לעומת זאת, מחפשת מיפוי קונפורמי מתחום המקור, ומנסה למזער את המרחק שלו אל תחום היעד. אנחנו משתמשים בקואורדינטות קושי־גרין כדי לייצג מרחב של העתקות קונפורמיות מתחום המקור, כאשר באמצעות ייצוג זה ניתן לחשב מיפויים קונפורמיים רציפים של תחום המקור בלי צורך לדגום את כל התחום, אלא רק את השפה שלו. כדי למזער את המרחק לתחום היעד אנחנו משתמשים באלגוריתם איטרטיבי המחפש אחר התאמה בין השפות של התחומים הקרובה ביותר להתאמה המתקבלת ממיפוי של תחום המקור על ידי אחד המיפויים הניתנים לייצוג על ידי הקואורדינטות. ייצוג זה מתבסס על אינטגרל המחושב על שפת התחום, המניב שיטה יעילה המאפשרת למשתמש לשנות אילוצים שונים המוגדרים על ידו בזמן אמת תוך כדי ביצוע האלגוריתם. באמצעות אילוצים אלו, הכוללים אילוץ של מיפוי "נקודה־אל־נקודה" או "סקיצה־אל־סקיצה", המשתמש יכול לשלוט על המיפוי ולבחור לאן יתמפו נקודות או עקומים מתחום המקור. בנוסף, אנחנו מראים איך השיטה ניתנת להכללה עבור מיפויים קוואזי־קונפורמיים, המאפשרים להעשיר את מרחב ההעתקות הניתנות לייצוג על ידי הקואורדינטות ולכן מתקבלת יותר גמישות המסתמנת באפשרות לספק את האילוצים שמגדיר המשתמש טוב יותר ולמצוא מיפויים עם פחות עיוות שטח.

שתי הבעיות שפתרנו בעבודה זו מנצלות את קואורדינטות קושי־גרין כדי להשיג ייצוג המבוסס רק על ערכים השמורים על השפה של התחום, וכתוצאה מכך האלגוריתמים הפותרים את הבעיות הם יעילים. לכן, מתאפשרת הוספת אינטרקטיביות עם המשתמש תוך כדי החישוב של אלגוריתמים אלו, המספקת למשתמש שליטה על התוצאה של האלגוריתם.

תקציר

הרבה יישומים שונים בגרפיקה ממוחשבת מערבים בעיה שנפרשת על פני תחום גדול ומעורבים בה הרבה משתנים. כתוצאה מכך האלגוריתמים שפותרים את הבעיה הם איטיים וכבדים, ולכן נמנעת האפשרות לממשק אינטרקטיבי בזמן אמת שמטרתו לאפשר למשתמש לשלוט על אופי הפתרון ולקבוע פרמטרים נוספים שחשובים עבורו. דוגמה ליישום כזה היא סימולציה של נוזלים. זהו יישום מאוד נפוץ בתעשיית הסרטים ליצירת אפקטים מיוחדים ואנימציות שמערבות נוזלים, כאשר הדרישות עבור הסימולציה הן שהיא תיראה אמיתית (ההתנהגות של הנוזל תראה נכונה לפי חוקי הפיזיקה) ושלאמן שיוצר את האנימציה תהיה יכולת לשלוט באופן שבו הזרימה תראה. בסימולציות נוזלים יש בדרך כלל צורך לעקוב אחרי התכונות של הנוזל (למשל, מהירות ולחץ) בכל הנפח שתופס הנוזל, על ידי ייצוג של הנוזל כאוסף של חלקיקים או באמצעות הגדרת רשת שהנוזל נמצא בתוכה כאשר בכל נקודה ברשת נשמרים הערכים של התכונות באותה נקודה. כתוצאה מכך, אלגוריתמים המשתמשים בייצוג זה נוטים להיות איטיים וכבדים, ולא מתאפשר חישוב ושליטה בזמן אמת על הזרימה. עבודה זו תעסוק במקרה מיוחד של סימולציות נוזלים שבו ניתן להקטין את עלות החישוב של הסימולציה באופן משמעותי על ידי ייצוג של תכונות הנוזל באמצעות ערכים הנשמרים רק על השפה של התחום של הנוזל.

התופעה שעבודה זו תעסוק בה היא יצירת "אצבעות" של הנוזל כתוצאה מהזרימה הצמיגה. בתופעה זו, כתוצאה מזרימה איטית של נוזל צמיג בתוך תווך עם תכונות מסויימות, נוצרות אי יציבויות בשפה של הנוזל שגורמות ליצירת תבניות מעניינות. ניתן לראות את התופעה הזו בטבע כאשר נוזל מחלחל לתווך נקבובי, והתבניות שמאפיינות אותה מופיעות גם בתופעות אחרות, כגון יצירת פתיתי שלג, גידול של בקטריות וגדילה מסתעפת. דרך נפוצה שבה חוקרים את התופעה היא באמצעות ניסוי שבו מזריקים נוזל פחות צמיג לנוזל יותר צמיג הכלוא בין שני לוחות מקבילים זה לזה וביניהם רווח קטן שמכיל את הנוזל. הניסוי הוצע ונחקר לראשונה על ידי הילי־שו, ועל כן נקרא בשם זרימת הילי־שו. כאשר הנוזל מוזרק באיטיות, מודל זה יוצר זרימה שבה מתפתחות במקרים מסויימים אי יציבויות בממשק שבין שני הנוזלים, שמובילות ליצירת האפקט של האצבעות, וניתנות לשליטה על ידי מספר פרמטרים של הזרימה כגון הצמיגות של הנוזלים, מתח הפונים בין שני הנוזלים וקצב הזרקת הנוזל.

היות והתבניות האלו היוו השראה להרבה אמנים ומעצבים, הם יוכלו למצוא שימוש בסימולציה של התופעה במחשב, המספקת בנוסף שליטה של המשתמש על תבניות האצבעות שנוצרות, תוך כדי שמירה על חוקי הפיזיקה ועל המראה של זרימה אמיתית. לצורך זה בעבודה זו פותח ייצוג של הבעיה המבוסס על אינטגרל שפה, המסתמך על התכונות המיוחדות של זרימה זו. ספציפית, זרימת הילי־שו היא זרימת פוטנציאל, שבה המהירות של הנוזל מונעת ישירות מהלחץ שלו והלחץ של הנוזל הוא פונקציה הרמונית. כדי לייצג את הפונקציה ההרמונית של הלחץ שמניעה את הזרימה השתמשנו בעבודה זו בקואורדינטות קושי־גרין הקומפלקסיות, שמאפשרות לפשט ולנתח את הבעיה ומספקות דרך יעילה ופשוטה לייצוג של הלחץ של הנוזל המתבססת על שמירת ערכים רק בשפה של התחום. אנחנו מראים בעבודה זו איך ניתן לשלב את קואורדינטות אלה בנוסחאות המתארות את זרימת הילי־שו, ואיך באמצעות ניסוח זה מתאפשר לסמלץ את הזרימה בזמן אמת ועל כן ניתן גם להוסיף שליטה של המשתמש על האופן שבו הזרימה תתנהג תוך כדי חישוב הסימולציה. בנוסף, אנחנו מאפשרים הזרקה או שאיבה של נוזל מעקום דיסקרטי, המורכב מאוסף של קטעים ישרים, ומראים איך ניתן ליישם את קואורדינטות קושי־גרין לתחום החיצוני של תחום סגור במישור או לתחום שאינו פשוט קשר (כלומר, לתחום המכיל מספר חורים). באמצעות ייצוג הלחץ בתחום החיצוני לתחום סגור

המחקר בוצע בהנחייתה של פרופסור מירלה בן־חן, בפקולטה למדעי המחשב.

חלק מן התוצאות בחיבור זה פורסמו כמאמרים מאת המחבר ושותפיו למחקר בכנסים ובכתבי־עת במהלך תקופת מחקר המאסטר של המחבר, אשר גרסאותיהם העדכניות ביותר הינן:

Aviv Segall and Mirela Ben-Chen. Iterative Closest Conformal Maps between Planar Domains. *Computer Graphics Forum*, 2016.

Aviv Segall, Orestis Vantzos and Mirela Ben-Chen. Hele-Shaw Flow Simulation with Interactive Control using Complex Barycentric Coordinates. *Proceedings of the 15th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2016.

תודות

אני רוצה להודות למנחה שלי, להוריי ולחבריי על התמיכה לאורך הדרך.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי.

סימולציות דו מימדיות ומיפויים באמצעות קואורדינטות קושי־גרין

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
מגיסטר למדעים במדעי המחשב

אביב סגל

הוגש לסנט הטכניון – מכון טכנולוגי לישראל
תמוז התשע"ו חיפה יולי 2016

סימולציות דו מימדיות ומיפויים באמצעות קואורדינטות קושי־גרין

אביב סגל