Lab on a Chip

PAPER

View Article Online
View Journal | View Issue

Cite this: *Lab Chip*, 2019, **19**, 3277

FlowSculpt: software for efficient design of inertial flow sculpting devices†

Daniel Stoecklein, ‡[b] Michael Davies,‡[a] Joseph Michael de Rutte,[b] Chueh-Yu Wu,[b] Dino Di Carlo*[b] and Baskar Ganapathysubramanian*[a]

Flow sculpting is a powerful method for passive flow control that uses a sequence of bluff-body structures to engineer the structure of inertially flowing microfluidic streams. A variety of cross-sectional flow shapes can be created through this method, offering a new platform for flow manipulation or material fabrication useful in bioengineering, manufacturing, and chemistry applications. However, the inverse problem in flow sculpting – designing a device that produces a target fluid flow shape – remains challenging due to the complex, diverse, and enormous design space. Solutions to the inverse problem have been constrained to single-material fluid streams that are shaped into top–bottom symmetric shapes due to the bluff-body structures available in current libraries (pillars) that span the height of the channel. In this work, we introduce multi-material design and symmetry-breaking flow deformations enabled by half-height pillars, presented within an extremely fast simulation method for flow sculpting yielding a 34-fold reduction in runtime. The framework is deployed freely as a cross-platform application called "FlowSculpt". We detail its implementation and usage, and discuss the addition of enhanced search operations, which enable users to more easily design flow shapes that replicate their input drawings. With FlowSculpt, the microfluidics community can now quickly design flow shaping microfluidic devices on modest hardware, and easily integrate these complex physics into their research toolkit.

Received 8th July 2019,
Accepted 26th August 2019

DOI: 10.1039/c9lc00658c

rsc.li/loc

1 Introduction

Shaping inertial microfluidic flow using sequences of pillar structures has recently gained traction in the microfluidic community as a method for passive fluid flow engineering. When laminar fluid flow with finite inertia ($1 < \mathrm{Re} < 100$, with the Reynolds number $\mathrm{Re} = \rho U D_h/\mu$, for fluid density ρ, characteristic velocity U, viscosity μ, and channel hydraulic diameter D_h) moves past a pillar in a microchannel, induced secondary flow alters the cross-sectional fluid shape in a time-invariant deformation to the structure of the flowing fluid.[1] Using this concept as a fundamental building block, a sequence of intelligently placed obstructions in a microchannel can produce a desired net flow deformation – a process we call "flow sculpting" (see Fig. 1(a)).

Flow sculpting enables new levels of passive flow control for a wide range of applications at the microscale. For example, although the fluid itself will be manipulated by a sequence of well-spaced micropillars, larger finite-sized particles may experience practically negligible deflection from their initial cross-sectional location in the stream.[1] This enables carrier fluid to be shifted away from focused streams of particles or cells, which can be leveraged for solution transfer or high-throughput extraction.[2] Advanced manufacturing processes are also enabled, with continuous-flow lithography creating multi-material polymers with 3D cross-sections, altering their mechanical or functional properties.[3] This can be extended *via* stop-flow like processes to create shaped 3D particles at millimeter[4] and micrometer[5,6] scales, which have applications in cell culture and analysis, tissue engineering, protein analysis, and additive manufacturing.[7] We envision future use of flow sculpting in biology and chemistry related applications by utilizing its unique ability to passively manipulate the cross-sectional distribution of fluid elements. For example, novel multi-material chemical concentration gradients could be generated in flow (or in UV-cured bio-compatible hydrogels) to study cellular chemotaxis.[8,9] Laminar flow reactions from co-flowing streams could be enhanced by optimizing micropillar sequences to maximize mixing while

[a] *Department of Mechanical Engineering, Iowa State University, Ames, Iowa, USA. E-mail: baskarg@iastate.edu*
[b] *Department of Bioengineering, University of California, Los Angeles, California, USA. E-mail: dicarlo@ucla.edu*

† Electronic supplementary information (ESI) available. See DOI: 10.1039/c9lc00658c

‡ These co-first authors contributed equally to this work.

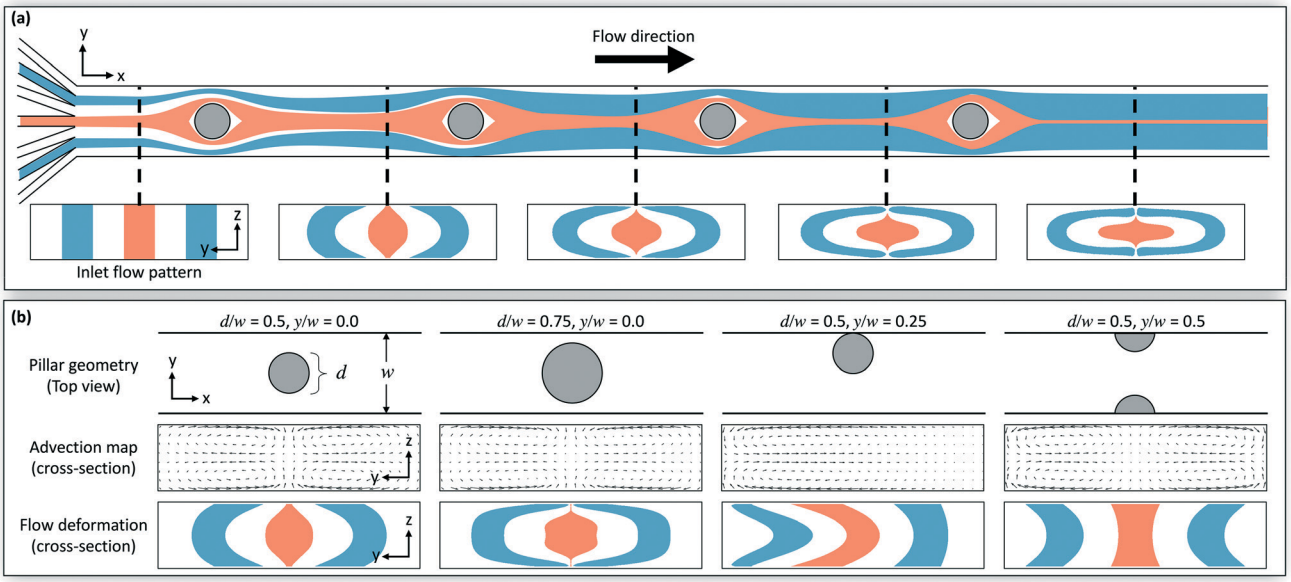

Fig. 1 Overview of flow sculpting. (a) A sequence of obstacles (pillars) deforms an inlet flow pattern (sculpted flow is colored blue and orange), with each obstacle contributing its own flow deformation to the overall net deformation at the outlet. In this illustration, the inlet flow pattern is a central stream of a width $w/5$ for a microchannel width w. (b) To accelerate flow sculpting device simulations, advection maps are created for a library of pillar geometries (shown is the advection map and flow deformation for a pillar of diameter d/w = 0.5 and lateral location y/w = 0.0), making whole-device simulation a matter of sampling stored 2D vector fields. Note the top–bottom symmetry in the advection map and flow deformations, due to the pillar spanning the height of the channel. For this figure, the channel aspect ratio is h/w = 0.25 and Re = 20.

minimizing pressure drop, potentially avoiding issues with clogging in standard chaotic micromixers.[10] Microfluidic biosensors could be designed that use a micropillar sequence to concentrate a rare target analyte toward a sensor embedded within a microchannel.[11]

Despite the apparent simplicity of incorporating inertial flow sculpting within a microfluidic device (adding cylindrical pillars), the difficulty and tedium in designing an appropriate pillar sequence makes its implementation more difficult in practice. Many current applications of flow sculpting were devised and implemented by inertial microfluidics experts, and even then, they sculpted relatively simple flow shapes that were achieved through trial-and-error design. A clear example of this difficulty is particle fabrication *via* flow sculpting, which remains competitive with state-of-the-art methods such as PRINT,[12] SEAL,[13] two-photon lithography,[14] and hollow fiber templating[15] due to the ability to create multi-material shaped 3D particles at high-throughput using basic microfluidic tools such as soft lithography[16] and 3D printing.[6] However, the flow sculpting approaches currently require the target 3D particle shape to comprise of the intersection of two orthogonally extruded 2D shapes: one from the shape of the sculpted flow stream (which contains a polymer precursor with a photoinitiator), and the other from an optical mask (which shapes polymerizing ultraviolet light). Such 3D particle shapes are thus limited not only to what is physically possible through inertial flow sculpting, but more practically, to a designer's ability to correctly arrange a sequence of pillars and inlet flow pattern to sculpt a desired cross-sectional flow shape. This highlights the broad necessity for a user-friendly tool to rapidly and accurately explore what flow deformations are possible, and aid multi-disciplinary, non-expert users in efficiently designing flow sculpting devices tailored to their needs.

Challenges typically encountered in modeling inertial flows (*i.e.*, nonlinearity in the governing equations, and fluid–structure interaction[17]) are significantly reduced in flow sculpting by the ease with which devices leveraging these physics can be simulated: if the distance between pillars is large enough to prevent cross-talk, each obstacle acts as an independent operator on the distribution of fluid elements within a channel cross-section. This allows for a library of pre-computed flow deformations known as advection maps (see Fig. 1(b)) to be rapidly concatenated for real-time flow simulation, as demonstrated in the freely available software "uFlow".[18,19] uFlow is a powerful visualization tool for manually exploring inertial flow sculpting, allowing users to rapidly build intuition with the complex design space. However, using uFlow to solve the inverse problem in flow sculpting – designing a flow sculpting device that produces a target fluid flow shape – is a monumental task given the enormous combinatorial difficulty, which is compounded by the nonlinear physics of flow sculpting.

Consider a library of 32 pre-computed deformations as used in uFlow in our previous exploratory work.[18] These 32 deformations were simulated for pillar geometries consisting of four diameters of $d/w = \{0.375, 0.5, 0.625, 0.75\}$ (for channel width w), and eight locations uniformly spanning the width of the channel, at Re = 20 and channel aspect ratio h/w = 0.25 (for channel height h). Complex fluid flow structures were

demonstrated for sequences up 12 pillars in length,[18] of which there are $32^{12} \approx 10^{18}$ possible pillar sequence configurations. The swirling, vortical nature of flow sculpting's advection maps further complicates design, where seemingly straightforward flow shapes may require an unintuitive pathway of intermediate structures that ultimately yields the desired result. Combined with the choice of how the fluid flow shape is configured at the microchannel inlet (called the inlet flow pattern, seen in Fig. 1(a)), a manual search for a particular flow deformation quickly becomes impractical. In fact, an analogous problem of choosing some set of nonlinear state transformations to match a desired net transformation has been shown to be NP-hard§ (nondeterministic polynomial time hard).[20] Hence, an automated – and possibly heuristic optimization – routine is required to make flow sculpting design accessible to the broader microfluidics community.

Heuristic searching methods have proven effective in solving inverse problems with discrete, non-differentiable, and multimodal design spaces.[21,22] In particular, the genetic algorithm (GA) has been shown to successfully optimize system geometry and operating conditions for fluid flow problems,[23–26] including flow sculpting.[27,28] The GA is based on the mechanisms of natural selection, where a population of candidate solutions randomly mutate and exchange design parameters according to stochastic selection and genomic modification processes, all attempting to maximize their individual fitness.[29] This technique was first employed for flow sculpting by Stoecklein *et al.*[27] to optimize pillar sequences for known fluid flow transformations, and search for arbitrary hand-drawn flow shapes. However, this framework had a very limited scope with considerable drawbacks: the application utilized commercial software (MATLAB) for deployment, users were required to specify the inlet design, the objective (fitness) function was a simple pixel-to-pixel image comparison on binary-valued images, and runtime was less than ideal at ≈24 hours per search. A 2nd generation GA framework for flow sculpting was created,[28] improving on some of these shortcomings with a freely available software "FlowSculpt". The FlowSculpt software included a customized GA framework that allowed for GA-designed inlet flow patterns and a reduced runtime of ≈2 hours, within a simple graphical user interface (GUI).

Still, the platform lacked sufficient features and utility to enable microfluidic researchers to employ flow sculpting in cutting-edge microfabrication approaches. Specifically, design was limited to single-material, top–bottom symmetric flow shapes. The ability to locally functionalize the surface of micro-particles or fibers with different materials is a significant strength of flow sculpting,[5] but with single-material design (*i.e.*, one color of sculpted fluid), users must first find a desired (single-material) shape, and then manually modify the inlet flow pattern to attain their intended material (color) distribution. Additionally, the use of pillars spanning the height of the microchannel enforced top–bottom symmetry in all flow deformations.¶ Runtime for the 2nd generation FlowSculpt is also quite lengthy at 2 hours, making rapid design iterations tedious and less approachable. This is exacerbated by the use of a pixel-to-pixel correlation objective function, which matches sculpted flow to the precise pixel locations in the target flow shape. This often leads to repeated design iterations, as a target flow shape may only be possible in a particular region of the cross-section, requiring the user to laterally shift target flow shapes based on the response from an initial guess. Finally, restricting flow to a single Reynolds number and aspect ratio can hinder the broad application of flow sculpting while constraining the design space: Amini *et al.*[1] established that varying either of these parameters (Re and h/w) can dramatically alter the flow deformation for a fixed pillar geometry (constant d/w, y/w), implying that a fully-featured tool for design in flow sculpting should allow variation of these terms.

In this work, we introduce multi-material design along with describing new asymmetric flow physics, and ameliorate all previously described shortcomings to the FlowSculpt framework. Simultaneous multi-material design is achieved with a new fluid state representation and objective function, and fully asymmetric flow sculpting is enabled by the use of half-height pillars, allowing deformations with both non-mirror symmetric vertical and lateral displacement of fluid. We couple this asymmetric design component to the FlowSculpt software with a new library of advection maps, including multiple channel aspect ratios and Reynolds numbers. We showcase fully asymmetric, multi-material inertial flow sculpting by using FlowSculpt with a continuous inlet flow pattern, demonstrating a variety of newly available flow stream topologies, and designing letters of the Roman alphabet from sculpted fluid, the latter of which we experimentally validate with confocal images of sculpted flow from fabricated devices. These advances are integrated with a 3rd generation of the FlowSculpt framework, which brings significant runtime improvements, design export functionality, and a rebuilt graphical user interface (GUI). We begin with a discussion of how devices are modeled within FlowSculpt, followed by a description of the new multi-material, symmetry-breaking flow physics, and asymmetric design capabilities. We then give an overview of how these components are implemented in the FlowSculpt software, and finish with benchmarks and experimental validation of the new flow physics.

§ The inverse problem in flow sculpting as defined in this work is in fact more difficult than that of ref. 20, as the target here is a scalar-valued result of an unknown (perhaps impossible) transformation, rather than the transformation itself. Here, it is unknown what state the flow must begin in, nor the route to their desired shape – only how we want the flow to appear at the end of a pillar sequence.

¶ Paulsen and Chung obviate this restriction by using mismatched densities in their sculpted streams.[30] This approach, while elegant, makes design much more tedious, and confines asymmetric flow sculpting to stopped-flow applications.

2 Modeling of flow sculpting devices

FlowSculpt's genetic algorithm relies on a forward model to generate a flow shape image for each candidate solution, which is evaluated within a fitness function – a process that is repeated many times throughout a single search. Hence, it is critical that the forward model accurately represents real-world device performance while executing quickly *in silico*. The forward model used in FlowSculpt computes net fluid displacement for a flow sculpting device by marching in sequence from pillar to pillar, sampling each pillar's advection map and displacing a set of cross-sectional fluid elements accordingly. Note that this method – and the entire scope of flow sculpting – is based on manipulating the cross-sectional structure of flowing single-phase Newtonian fluid, and does not predict how immiscible materials or finite-sized particles will behave. This method of device simulation has been well-validated in our previous work.[1,18,27] Here, we significantly improve the speed of FlowSculpt's forward model (by a factor of 34) without loss of accuracy, and enable multi-material design, by the use of index maps. We first discuss advection map generation and the forward model, and then describe the implementation of index maps.

2.1 Advection map generation

Advection maps are 2D vector fields describing the net lateral displacement of fluid flowing through a 3D microchannel domain (see Fig. 1(b)). Previously, FlowSculpt used advection maps created from simulated 3D domains containing pillars spanning the height of the microchannel, with normalized diameter d/w and location y/w for microchannel width w. In theory, any structure which induces secondary flow can be used, as long as it conforms to the constraints of flow sculpting. These constraints – necessary for accurate predictions of sculpted flow in our forward model – are enumerated here:

1. Each structure-induced flow deformation must be completed before the fluid begins to interact with neighboring pillars (*i.e.*, no cross-talk).

2. The structure-induced flow deformation must not vary in time (*e.g.*, no periodic motion or shed vortices).

3. For a set of obstacles to be used in a flow sculpting sequence, each neighboring obstacle's domain must match flow conditions (Re) and channel geometry (h/w).

These rules for flow sculpting allow a broad set of usable flow physics and geometry. To generate advection maps, we solved the 3D Navier–Stokes equations using our experimentally validated in-house finite element method (FEM) software[18,27] and the Gmsh meshing software[31] for a multitude of microchannel and pillar geometries, and Reynolds numbers, as described below. Each simulated 3D domain has $\geq 2d/w$ of empty channel spacing upstream of the pillar, and $\geq 6d/w$ spacing downstream, which allows flow deformation to saturate before exiting the domain.[1] The resulting 3D velocity fields are then streamtraced with neutrally buoyant, infinitesimal particles, yielding 2D cross-sectional fluid displacements – an advection map – for each flow sculpting component.

2.2 The forward model

The forward model predicts how a sequence of flow sculpting components will deform an inlet flow pattern (see Fig. 1(a)). Computing the flow deformation caused by each sculpting component starts with a costly 3D CFD simulation (described above) which is reduced to a 2D advection map. The forward model then computes fluid displacement by sampling a library of advection maps in sequence. Stoecklein *et al.*[27] further reduced the complexity of flow shape simulation by discretizing the microchannel cross-section into a set of cells representing fluid elements as binary states (either 1 for sculpted flow, or 0 for co-flow). In this scheme, advection maps are converted into sparse transition matrices which describe how fluid will transition from one discretized cell to another.[27,28] Simulating a flow sculpting device then becomes a matter of linear algebra, with matrix–matrix multiplication producing a net transition matrix, which can be multiplied by a 1D vector representing the (reshaped) inlet flow pattern, returning a 1D vector representing the outlet fluid states. The vector can be reshaped back into a 2D matrix and processed as an image, with each fluid cell being a pixel.

This approach is easily ported to various CPU or GPU architectures, and readily pipelined directly into optimization routines. However, a tradeoff is made between speed and accuracy based on the number of fluid cells in the discretized cross-section. For a coarse discretization of fewer (but spatially larger) cells, the number of linear algebra operations are minimized, thus making for a faster forward model. Larger fluid cells could mean a significant loss of information in the advection data, as the characteristic size of the cells determines the degree to which more subtle fluid flow displacement is captured in the transition matrix. On the other hand, a high resolution discretization with smaller fluid cells will retain more information, but at the cost of additional operations during flow sculpting simulation. We previously[27] conducted a study of discretization resolution by comparing information degradation *vs.* speed for the advection maps used in FlowSculpt, and found that for microchannel aspect ratio $h/w = 0.25$, a resolution of $N_Y = 800$ cells along the width of the channel and $N_Z = 200$ cells along the height gives a satisfactory tradeoff between accuracy and speed.

There are additional tradeoffs in describing continuous fluid displacements – each having a unique time-of-flight through the 3D domain – using a discrete grid. Consider fluid streamlines in a forward direction from inlet-to-outlet in a 3D domain containing a flow-deforming pillar: some fluid parcels at different inlet cells will displace to the same cross-sectional cell at the outlet, due either to information loss in converting from continuously-valued advection to a discrete transition, or different fluid arrival times. This potential for a many-to-one mapping results in an apparent

increase in the density of tracked fluid at destination cells, while "empty" cells are formed in neighboring regions, creating a pockmarked image that poorly represents a sculpted fluid's real-world continuous shape, and making image comparison algorithms less useful. In our previous work,[27] we used outlet-to-inlet streamtraces (reverse advection) to create a one-to-one mapping that "pulls" fluid from a cross-sectional distribution at the inlet into a uniformly discretized grid at the outlet. This creates flow shapes with a continuous (filled) shape, but allows fluid to displace from one cell at the inlet into two (or more) cells at the outlet, potentially increasing the apparent fluid mass. However, since our overall goal is based on flow shape comparisons to a target image, the loss of perfect mass conservation is not at all detrimental when compared to the boost in image quality (though other measures that consider material concentration, for example, would suffer).

It is important to note that the CFD simulations that create the advection maps used in FlowSculpt's forward model are non-dimensional, scaling by the Reynolds number, Re, using the channel hydraulic diameter D_H as the characteristic length. Therefore, the predictions of the forward model (and designs created by FlowSculpt) can be used at any lengthscale where the relevant physics are matched. That is, Re should match for Newtonian fluids used in simulation and experiment, and the Péclet number Pe = UD_H/D (with material diffusivity, D), which represents the balance between advection and diffusion in flow, should be large enough reflect the diffusion-free predictions from the forward model.

2.3 Fast index maps and multi-material design

In this work, the fluid advection operation is further reduced into a one-to-one index mapping. Rather than using the matrix–vector multiplication scheme from our previous forward models, we leverage the one-to-one mapping inherent to reverse advection streamtracing to create index mappings, which relay the same displacement information as a transition matrix, but in a simple 1-dimensional (1D) vector. For a set of uniformly discretized cells $A = \{a_i\}$ in an inlet cross-section (flattened into a 1D array), and the corresponding output cross-section $B = \{b_j\}$, we define an index mapping f: $A \rightarrow B$ that relates an output cell index j in B to an input cell index i in A. A transition for a given mapping f and input cross-section a is performed by computing

$$b_j = a_{f(j)}$$

For all b_j in the outlet cross-section. The function f is one-to-one since each output index has exactly one input index. Fig. 2 shows this mapping visually.

Index mappings provide the same black-box forward model the previously used transition matrices, but with less computation. Instead of performing matrix–matrix multiplication, which in general has a worst case runtime complexity greater than $\mathcal{O}(n^2)$,[32] the index mapping method is essentially a lookup table, with a runtime complexity[33] of $\mathcal{O}(1)$. Moreover, the use of a lookup table over matrix–matrix multiplication allows for arbitrary value types to be used for fluid states, enabling simple multi-material design. We now allow integer values 0 through 5 to populate the fluid states, where each value is considered a separate neutrally buoyant material within the fluid. We emphasize that this is not modeling a multi-phase system, and that each material used should have similar physical characteristics (*e.g.*, miscibility, viscosity, and density), as the forward model treats each material equally.||

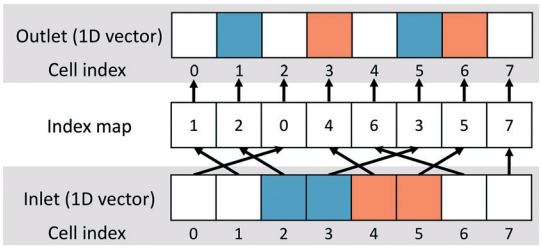

Fig. 2 Diagram depicting the operation performed by an index mapping on a 1D vector of eight fluid states (colored blue and orange to show tracked fluid) from inlet to outlet. In addition to significantly improved speed, index maps enable simple multi-material design.

This kind of similar-property, multi-material flow sculpting has been demonstrated in ref. 5 and 16, where adjacent flow streams of poly(ethylene glycol) diacrylate and poly(ethylene glycol)-acrylate-biotin were sculpted to create 3D shaped microparticles with biotinylated surfaces, enabling functionalization *via* streptavidin-linked functional groups (*e.g.*, incubating biotinylated microparticles with streptavidin and biotinylated collagen to create microcarriers for collagen-receptive cells[16]). In ref. 5, two materials of varying viscosity (Norland Optical Adhesive 89 and triethylene glycol dimethacrylate) were co-flowed, creating microparticles with some small error in comparison to their predicted flow shape, hinting that some discrepancies between material viscosity can be allowed. In these works, the flow shapes were designed in a manual fashion, either by adjusting the pillar sequence and inlet flow pattern simultaneously using the uFlow software,[5] or by first finding the desired flow shape, and then tuning the inlets in the hopes of preserving the desired spatial distribution of material in the final sculpted flow shape.[16] FlowSculpt's multi-material model does away with this tedium, simultaneously optimizing flow shape and material distribution in a fast, automated framework.

2.4 The FlowSculpt library

FlowSculpt's advection maps previously sampled from a library of pillars of a height $h_p/h = 1.0$ (for pillar height h_p and channel height h) at a fixed channel aspect ratio $h/w = 0.25$,

|| Fluids are assumed to be nearly identical, with negligible interfacial tension.

and fixed Re = 20, thus limiting cross-sectional flowshapes to have symmetry across the horizontal mid-plane (see Fig. 1), and restricting the flow physics based on Re and channel confinement. Below, we describe FlowSculpt's expanded set of flow transformations and their impact on the design space.

2.4.1 Half-pillars and additional Reynolds numbers. We break horizontal mid-plane flow shape symmetry by adding partial-height pillars for which h_p/h = 0.5 (see Fig. 3) to FlowSculpt's library of flow deformations. While additional pillar heights would provide more diversity in the design space, we focused only on half-height pillars for now, in order to simply break symmetry while retaining a degree of control over local flow disturbances akin to full-pillars. Moreover, mixing-and-matching pillar heights in multi-stage lithography greatly increases the complexity in fabrication, though it could be more easily accomplished as 3D printing precision improves. We define the two pillar geometries as full-pillar (h_p/h = 1.0) and half-pillar (h_p/h = 0.5). For each type, we simulated 3D flow fields for pillar locations spanning y/w = [0 : 0.125 : 0.5], with y/w = 0 being the center of the microchannel (pillars for which $y/w < 0$ simply mirror their counterparts across y/w = 0). Pillar diameters for full-pillars were d/w = {0.375, 0.5, 0.625, 0.75}, while half-pillars used the same set, but appended with two larger diameters d/w = {0.875, 1.0}. Four different Reynolds numbers were computed (based on hydraulic diameter, as defined prior in the text) for each geometry: Re = {10, 20, 30, 40}. Analysis of deformation saturation length L_s for half-pillar simulations determined the same guidelines of full-pillars, which require downstream inter-pillar spacing $L_s \geq 6d/w$.

Secondary flows induced by half-pillars tend to affect fluid primarily in the lower-half of the channel near the pillar. However, full-width (d/w = 1.0, depicted in Fig. 3) half-pillars force all of the fluid into the top-half of the channel as it flows past the pillar, with fluid entering the bottom-half of the channel first at the point where the pillar's geometry is tangent to another surface – either the channel walls (for y/w = 0.0 – see Fig. 3(a)), or where the pillar meets its periodic counterpart (*e.g.*, in the middle of the channel for y/w = 0.5 – see Fig. 3(b)). In examining the d/w = 1.0, y/w = 0.0 geometry, the full-width pillar's edge is tangent with both channel walls, allowing fluid to re-enter the lower half of the channel near the walls before fluid passes over the center-line edge of the pillar, which creates a secondary flow forcing fluid from the walls to the center of the channel, and then up toward the top-wall of the channel (see the advection map and flow deformations in Fig. 3(a)). Conversely, for d/w = 1.0 and y/w = 0.5 (with periodic geometry) fluid re-enters the lower half of the channel in the center, with its inertia forcing displacement out toward the walls (Fig. 3(b)). Overall, these full-width half-pillar flow deformations introduce a powerful set of transformation to the flow sculpting toolkit, for which we show experimental validation in Fig. 3. As we will see, these effects are diminished in higher aspect-ratio channels, but at h/w = 0.25, the unique forcing of fluid toward the upper or lower surfaces of the channel could be exploited for enhancing the efficiency of sensors embedded in the microchannel,[34,35] for example.

2.4.2 Increased channel aspect ratios. Preliminary work in flow sculpting showed that varying channel aspect ratios and Re will alter the mode of advection, changing the distribution and strength of induced secondary flows.[1] FlowSculpt now gives access to multiple channel aspect ratios h/w = {0.25, 0.5, 1.0} (for channel height h and width w) and Re = {10, 20, 30, 40}. A variety of deformations are shown in Fig. 4, where pillar geometries using the same blockage ratio β = 0.5 are compared with two lateral offsets (y/w = {0, 0.5}) and different aspect ratios for full-pillars in Fig. 4(a and b), and half-pillars in Fig. 4(c and d).

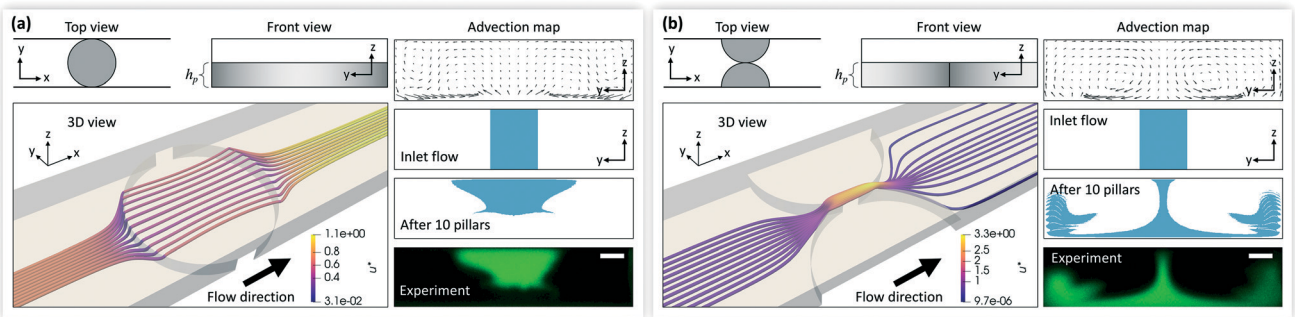

Fig. 3 Top–bottom asymmetric fluid flow transformations from half-pillar obstacles. (a) A full-width (d/w = 1.0) half-pillar (h_p/h = 0.5) placed at y/w = 0.0 will drive fluid in the center of the channel upward, resulting in central flow streams being displaced to the upper-half of the microchannel. The 3D view shows streamtraces originating from a flat line from y/w = −0.1 to y/w = 0.1 at z/h = 0.04, colored with a non-dimensional velocity $u^* = u/U$, for downstream velocity u and characteristic velocity U. Note the post-pillar velocity increase, due to the streamtraces being advected into a faster region closer to the center of the channel. This is seen in the advection map, and the deformation of an inlet flowpattern using a sequence of 10 such half-pillars. (b) Full-width half-pillars at y/w = 0.5 (with periodic pillar geometry) create a flow transformation complementing that of y/w = 0.0, with fluid being forced into the lower-half of the microchannel near the channel center, seen in the 3D view streamtraces and the resulting advection map, along with the deformation of the same inlet flow pattern used in (a) from a sequence of 10 half-pillars. We validate these predictions with confocal imaging from experiments (see Methods section). The scale bar is 20 μm.

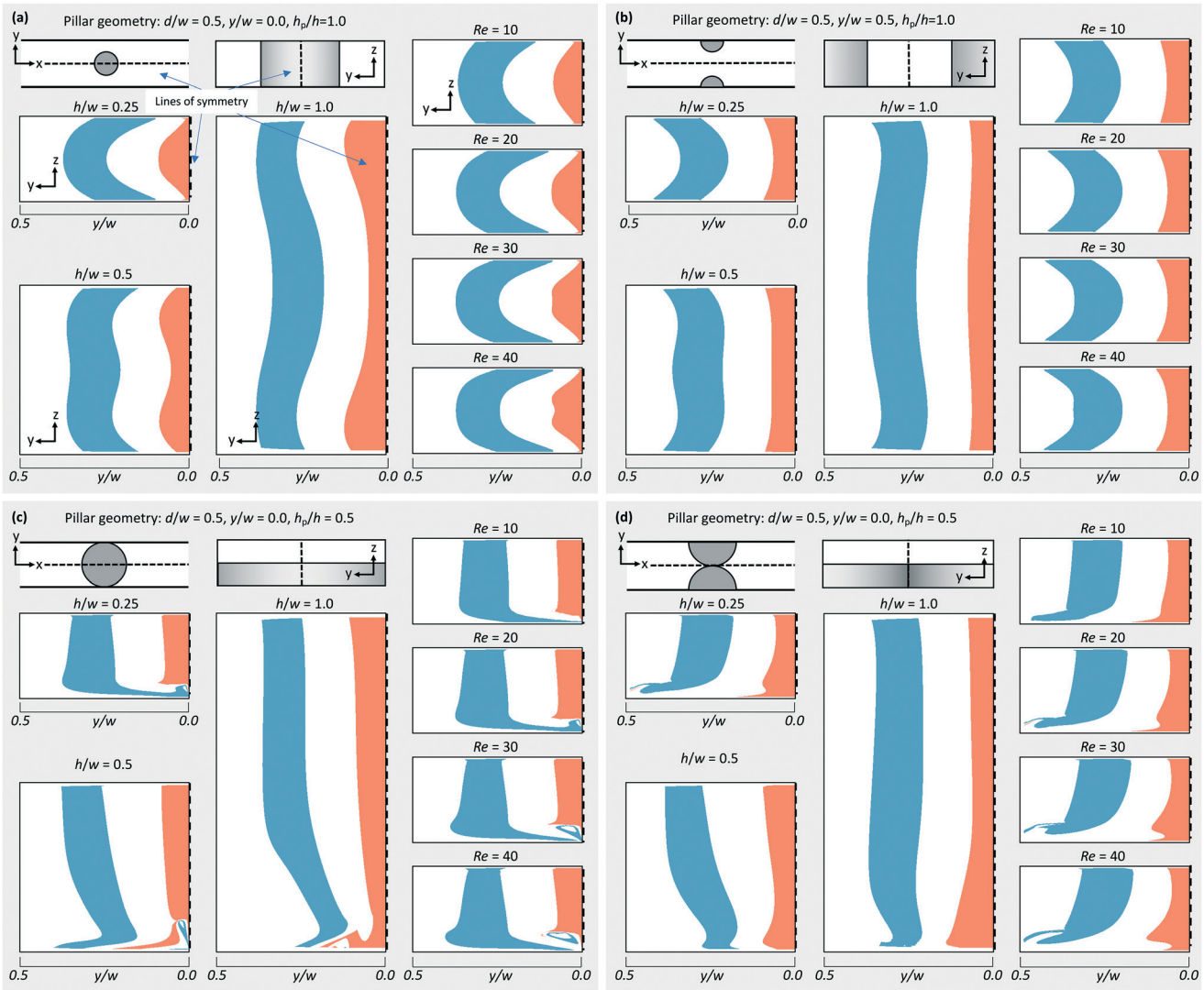

Fig. 4 We use our forward model to demonstrate the variety of sculpted flow shapes arising from changing channel aspect ratio h/w, pillar location y/w, pillar height h_p/w, and flow Reynolds number Re, by transforming the same inlet flow pattern depicted in Fig. 1. Four pillar geometries are compared: full-pillars (h_p/w = 1.0) (a) d/w = 0.5, y/w = 0.0, (b) d/w = 0.5, y/w = 0.5; and half-pillars (h_p/w = 0.5) (c) d/w = 1.0, y/w = 0.0, (d) d/w = 1.0, y/w = 0.5. These geometries have the same blockage ratio, and are symmetric across the vertical centerline of the channel. For brevity, we show only the left-hand (+y) side of the flow deformation, using dashed lines to indicate the plane of symmetry. For each geometry, three channel aspect ratios h/w = {0.25, 0.5, 1.0} are shown for Re = 20, and four Reynolds numbers Re = {10, 20, 30, 40} are shown for h/w = 0.25 (a commonly used aspect ratio due to relative ease of fabrication).

We observed that as the aspect ratio of the channel increases, stronger flow displacements tend to be more localized to where the pillar meets the walls at the top and bottom of the channel (or the side walls, for off-center pillars), in contrast with the lowest aspect ratio of h/w = 0.25, where the fluid displacement magnitude is more uniformly distributed throughout the channel cross-section. This influence of confinement on local velocity gradients correlates with observations of inertial flow past a 3D confined cylinder by Ribeiro *et al.*,[36] who showed that a higher aspect ratio channel (low confinement) increasingly separates the end-wall sources of vorticity from the interaction of the fluid with the pillar alone near the symmetric mid-plane. For half-pillars, increasing the channel aspect ratio does modify how the fluid is sculpted, but most of the deformation is within the lower-half of the channel near the half-pillar, occurring closer to the channel floor as confinement decreases.

For each pillar geometry, the same lateral offsets are compared for Re = {10, 20, 30, 40} at h/w = 0.25. As expected, increasing Re tends to exaggerate characteristic features of the flow deformation, eventually introducing new features to the sculpted form. In general, higher Re seems to enable a more diverse and capable flow sculpting design space.

3 FlowSculpt software methods

Solving inverse problems related to fluid–structure interactions usually requires high-performance computational

resources that preclude widespread adoption by non-experts or users with modest computing power. Instead, we embed flow sculpting's fast forward model within an evolutionary algorithm (the GA), and make automated flow sculpting design easily accessible *via* the FlowSculpt software.** In the following sections, we describe FlowSculpt's optimization process, fitness functions, and GUI usage.

3.1 Genetic algorithm (GA) optimization

We quickly outline the GA and describe how it can solve the inverse problem in flow sculpting. The GA is initialized by generating a random population of candidate solutions known as chromosomes, each of which contains a standalone design for the problem of interest. The population of chromosomes is then evaluated using a "fitness function" (objective function), which returns a scalar value representing the fitness of each chromosome, with higher fitness being more desirable. Fitness is used to determine the next generation of chromosomes by a stochastic selection process, which chooses parents from the current generation to exchange components of their design variables *via* crossover to form offspring for the next generation. Higher fitness is rewarded with a greater chance of selection, thus propagating genetic material associated with successful design throughout the population. Conversely, chromosomes with poor fitness are less likely to be selected, and their designs will tend to wash out of the population. Chromosomes are also randomly selected to have their genetic material perturbed (mutated), and placed in the next generation. The best-performing chromosomes are chosen as elites and passed on untouched to the next generation, thus preserving a running optimum solution. The population will then step forward, repeating the processes of evaluation, selection, crossover, and mutation to create a new generation of chromosomes. This evolutionary step is repeated until some termination criteria are met, and the best-performing chromosome is output as the solution. The GA is a stochastic process, so finding the global optimum is not guaranteed; therefore, it is important to re-run the GA some number of times (usually at least 10 times) in order to attempt statistical exploration of the space.

There are two essential components of the GA to be implemented to solve an inverse problem: a chromosome containing problem-specific design parameters as genetic material, and a fitness function to evaluate each chromosome during the GA's evolutionary step.

3.1.1 GA chromosome. Flow sculpting device design is encoded into a chromosome as two separate parts: an inlet flow pattern and a pillar sequence. The inlet flow pattern in this work departs from previous binary-valued descriptions,[27,28] instead using a continuous-valued design for the flow rate fractions, where locations σ_i for $i = \{1, \ldots, n\}$ delineate $n + 1$ boundaries between the co-flows, with accompanying integer-valued material indicators S_j for $j = \{1, \ldots, n + 1\}$ materials (see Fig. 5). This description of the inlet flow pattern is a more appropriate match for real-world microfluidic devices, in which co-flow fractions at the inlet can be controlled by continuously tuning inlet flow rates. In this scheme, the number of possible inlet volumes and material types must be supplied (currently limited to 9 inlets and 6 different materials), and a minimum flow cross-section size must be enforced to maintain inlet flow patterns which can be reasonably achieved in practice. The pillar sequence is described in the chromosome as a fixed-length real-valued sequence with separated diameter and location values for each pillar. To account for half-pillars, we add an extra component to each pillar's design to select the pillar height.††

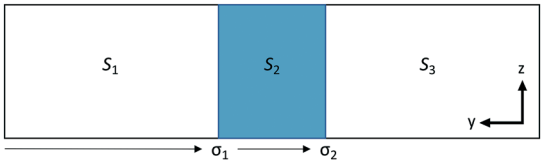

Fig. 5 FlowSculpt's inlet design uses continuous-valued parameters σ_i to describe the inlet flow co-flow cross-sectional fractions, while each co-flow's material is described by the integer-valued S_j. Here, three co-flows containing materials S_1, S_2, and S_3 are separated by boundaries at σ_1 and σ_2.

3.1.2 GA fitness function. FlowSculpt's default pixel-to-pixel fitness function employs the same correlation coefficient $r(I_{sim}, I_T)$ as used in ref. 27 and 28, which compares a chromosome's simulated flow shape image I_{sim} to a target flow shape I_T. The correlation function r is a measure of similarity between the two images I_{sim} and I_T, with a value of $r = 1.0$ meaning that both images match pixel-to-pixel. For multi-material design, we use a modified correlation function which only counts matched materials, and report the average result as a chromosome's fitness.

To allow designs for which precise flow shape location is not important, we have included a translation invariant fitness function (which is currently valid for single-material design only). We have also provided hooks within the FlowSculpt code to easily access the open source computer vision library (OpenCV),[37] which has many image processing routines, thus making the creation of customized fitness functions a more simple task. These implementations are detailed next.

3.1.2.1 Translation invariant fitness function. To provide a translation invariant fitness function which searches for a target flow shape at any translated position within the channel, we exploit the "shift" property of the Fourier transform, whereby the translation of an image will be

** Binary executables for FlowSculpt on Windows, MacOS, and Linux, as well as its source code, are freely available at www.flowsculpt.org.

†† Because the top-half of the channel is completely open in half-pillar geometries, larger diameters of $d/w = \{0{:}875, 1{:}0\}$ can be utilized without drastically increasing pressure drop. We therefore constrain the chromosomes to prevent full-height pillars from attaining these larger, infeasible diameters.

reported as a phase shift in frequency space, while the magnitude of the transforms will remain the same.[38] Within this fitness function, we compute the fast Fourier transform (FFT) of the target I_T and chromosome-produced simulated image I_{sim}. By comparing the magnitude of each image's FFT we introduce translation invariance to the measure of fitness, thereby searching for a fluid flow shape without regard to where it is located within the microchannel cross-section. In FlowSculpt, this is accomplished by using the previously described correlation function r on the difference between the images' magnitudes in Fourier space, which are computed using FFTW:[39]

$$\text{fitness} = r\left(\left|\mathcal{F}\left(I_{sim}\right)\right|, \left|\mathcal{F}\left(I_T\right)\right|\right) \quad (1)$$

3.1.2.2 Custom fitness functions and OpenCV. We have separated the implementation of fitness functions from FlowSculpt's primary routines as a standalone source file fsfitness.cpp. This allows users to specialize their flow sculpting design optimization based on domain knowledge and problem-specific criteria. As an additional aid to researchers designing their own fitness functions, we have also added support for the OpenCV library. OpenCV is a freely available computer vision library that contains a multitude of techniques for image processing and comparison. We anticipate our future publications will explore OpenCV methods for flow sculpting, but researchers can begin utilizing it in their own work immediately.

3.2 Graphical user interface

We have embedded the FlowSculpt software (written in C++) within a GUI using the cross-platform application framework Electron.‡‡ The FlowSculpt app which uses the GUI (shown in Fig. 6) is freely available at www.flowsculpt.org, and can be easily installed on Windows, MacOS, and Linux operating systems.

The top section of the GUI is dedicated to user input for the target flow shape, with a drawable canvas and modifiable drawing brush. There are also entries for flow conditions (Re), channel geometry (h/w), and a selection to enable half-pillars. If half-pillars are disabled (full-pillar design only), anything drawn in the top-half of the canvas will be automatically mirrored in the lower-half, and *vice versa*; otherwise, the entire canvas is open to shape design. Finally, in addition to a button to clear the drawing canvas, the "Load image" button allows users to import target flow shapes from images made outside of FlowSculpt. Here, the user should be aware that some images will have aliasing or compression effects on the edges of shapes, creating additional colors within the image. FlowSculpt will use a nearest-neighbor algorithm to flatten these colors, but there is no guarantee of matching the user's intent.

‡‡ https://github.com/electron/electron.

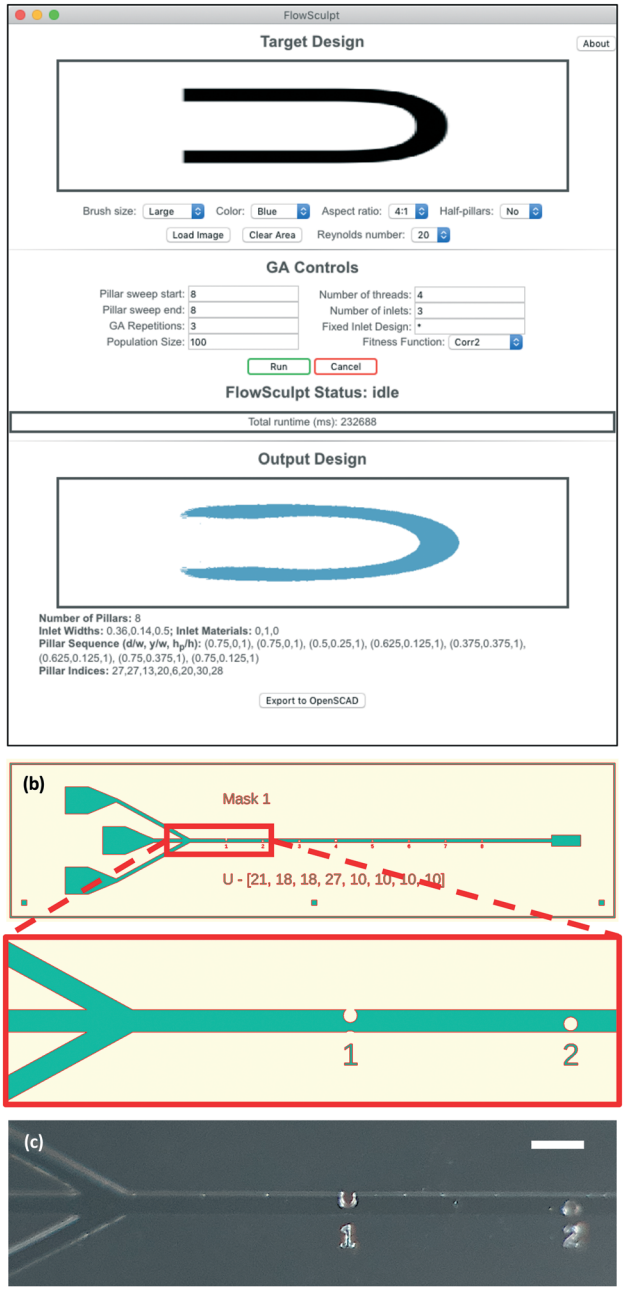

Fig. 6 (a) The graphical user interface (GUI) for FlowSculpt, created using electron. The top section (target design) contains a drawing canvas for the user-designed target flow shape, with adjustable marker size and fluid material color. By default, FlowSculpt will use pillars spanning the height of the channel, forcing symmetric design within the canvas. By choosing to use half-pillars, the entire canvas can by drawn on. The middle section (GA controls) configures parameters for FlowSculpt's optimization. The lower section contains the results of the last search, and a button to export the design to OpenSCAD. (b) An example output from FlowSculpt to OpenSCAD, with a zoomed-in portion of the device showing the inlet junction and individual pillars with numbered labels. (c) An image of a PDMS device fabricated from the OpenSCAD design in (b) using soft lithography (see Methods section). The scale bar is 500 μm.

The middle section of the GUI configures runtime parameters for FlowSculpt: pillar sweep start/stop will choose the

allowable number of pillars; GA repetitions determines how many times the GA is repeated per pillar sequence length; population size controls the GA population; number of threads chooses how many threads (*i.e.*, CPU cores) to use during parallel evaluation; number of inlets limits how many channels the GA can use to create an inlet flow pattern; fixed inlet design allows the user to force a particular inlet flow pattern (*e.g.*, [0, 0.4, 0.2], [0, 1, 0] for a single sculpted stream in the center one-fifth of the channel); fitness functions can be selected from a small menu. The default values for each configuration are sufficient to provide a quick result within a well-resolved design space (4–8 pillars), but users are encouraged to modify these values to ensure a more exhaustive search that is tuned to their experimental constraints. The lower part of this section contains buttons to run FlowSculpt on the target design or cancel a run that has started, as well as a window that provides updates on the status of FlowSculpt as it is working.

The bottom section of the GUI gives the resulting design from FlowSculpt optimization, along with the number of pillars used, inlet flow pattern, and pillar sequence. The pillar sequence design can be exported to the open source computer-aided design (CAD) software "OpenSCAD"§§ (see example output in Fig. 6(b)). Devices using half-pillars will automatically generate two masks for two-stage soft lithography, which we use for creating pillars of different height.

4 Experimental methods

4.1 Fabrication

For verification of devices designed in the following section, microfluidic channels incorporating the designed pillar sequences were fabricated from the CAD drawings exported from FlowSculpt using soft lithography. The molds corresponding to the channel design were created on a silicon master coated with two 25 μm layers of KMPR 1010 (MicroChem Corp.) to form devices with cross-sections of width × height of 200 μm × 50 μm, each patterned in sequence using standard photolithography techniques. Polydimethylsiloxane (PDMS) base and curing agent (Sylgard 184 Elastomer Kit, Dow Corning Corporation) were mixed at a ratio of 10 to 1, poured onto the molds in petri dishes, put in vacuum to remove bubbles, and cured in an oven at 65 °C to replicate the structure of the microchannels. The PDMS devices were peeled from the mold and punched with holes at the inlets and outlets. The PDMS devices and glass cover-slips (no. 1.5, Electron Microscopy Sciences) were then activated *via* air plasma (Plasma Cleaner, Harrick Plasma) and bonded together to enclose the microchannels. The PDMS devices were then infused with rhodamine B (0.1 mg mL^{-1}, Sigma-Aldrich) to help visualize the channel walls.

§§ www.openscad.org.

4.2 Confocal imaging

Confocal images of the fluid flow deformation for optimized designs were taken downstream of the fabricated pillars using a Leica inverted SP5 confocal microscope at the California NanoSystems Institute. For each design, syringes on separate syringe pumps (Harvard Apparatus PHD 2000) were connected to the inlets of the microchannel using PEEK tubing (Upchurch Scientific product no. 1569). We visualized sculpted flow streams using deionized (DI) water with fluorescein isothiocyanate dextran 500 kDa (5 μM, Sigma-Aldrich) while co-flow streams contained DI water only. The total volume flow rate was 7.5Re μL min^{-1} (based on microchannel hydraulic diameter), with the flow rate of each inlet stream configured to its cross-sectional area before the first pillar in the design. Confocal images of the cross-sectional planes were taken at locations approximately four times the channel width w downstream of the pillars. For each measurement, random noise was eliminated by averaging six images taken over 3 second intervals to arrive at a final image.

5 Results and discussion

We tested the FlowSculpt framework in two campaigns: (1) benchmarking and (2) flow shape demonstration. The benchmarks are intended to showcase the speed of the index maps and the tradeoffs of translation invariant design. For the demonstration of flow shapes, we used FlowSculpt to design several multi-material and asymmetric flow shapes, including letters of the Roman alphabet for which we fabricated some corresponding devices to visualize the sculpted flow, and used confocal imaging for validation.

5.1 Benchmarks

We recorded results for benchmark images with several test cases to evaluate the modifications to FlowSculpt. All tests were run on a system with a 6-core (12 thread) Core i7 4960X processor and 32 GB of memory running Ubuntu 16.04. The cases ran were the following:

Case study 1: index map runtime A speed comparison of the forward model using index maps model *vs.* matrix multiplication.

Case study 2: translation invariant runtime testing the relative speed of the new GA with and without an FFT pass for translation invariance.

Case study 3: translation invariant design evaluating the benefits of translation invariance on artificially perturbed flow shapes.

5.1.1 Case study 1: index map runtime. A set of 60 randomly generated target flow shapes were used to compare FlowSculpt's forward model using index maps *vs.* matrix-multiplication, as used in ref. 27 and 28. These flow shapes are the same set used in ref. 28. The purpose of this test was to compare the use of transition matrices (using matrix–matrix multiplication) to the index maps introduced in the present work. We used the same parameters for both GA setups aside

from the modified fitness evaluation. GA parameters were a population size of 100, sweeping from 5 to 10 pillars with 10 repetitions of the GA at each step (for a total of 60 GAs per flow shape search). For all runs in case 1, we used 800 × 100 resolution cross-sections (half-channel, for h/w = 0.25), and fixed the inlet flow pattern.

Results are shown in Fig. 7(a). The speedup from the use of index maps is significant, with no practical difference in accuracy. Where matrix–matrix multiplication required, on average, ≈5000 seconds (1.4 hours) per image search, index maps required ≈145 seconds (2.5 minutes), on average. This is a 34× reduction in runtime, with the same quality of results. Such drastic improvements in forward model runtime are critical for optimization problems, as they allow for overall optimization routines to favor coverage of the search space with little to no deference to runtime. In FlowSculpt, this means a single flow shape search can use hundreds of GA repetitions per pillar sequence length, in the same time that only 10 repetitions were previously used. A faster forward model also provides additional breathing room in the fitness function pipeline to implement computationally costly methods, such as the translation invariant FFT post-processing presented in this work.

5.1.2 Case study 2: speed comparison with translation invariance. Here, we analyzed FlowSculpt runtime and GA convergence for the translation invariant fitness function. We use the original set of 60 target flow shapes from case study 1 in order to isolate differences in performance to the FFT-based function's difficulty in searching a translation invariant design space. That is, both fitness functions are equally capable of finding exact solutions to these target flow shapes, but the FFT-based fitness function will have a more difficult search.

Fig. 7(b)(i) shows the number of generations needed for GA convergence across the entire test set, while Fig. 7(b)(ii) shows a comparison of FlowSculpt runtime with and without translation invariance. Both fitness functions find good results for each target flow shape. However, FlowSculpt's translation invariant searches required more generations to converge, with an average of 120 generations until convergence with FFT *vs.* 89 generations without FFT. This is expected due to the larger and more diverse translation invariant search. Use of the FFT increases FlowSculpt runtime by a factor of ≈3, with an average FlowSculpt runtime of ≈442 seconds with FFT *vs.* ≈145 seconds without.

Despite the increase in runtime in using the FFT-based fitness function when compared to the correlation function, conducting a translation-invariant search using the FFT fitness function avoids a campaign of manually-translated target flow shapes. In fact, for these test images, a user need only attempt four such manual translations before their

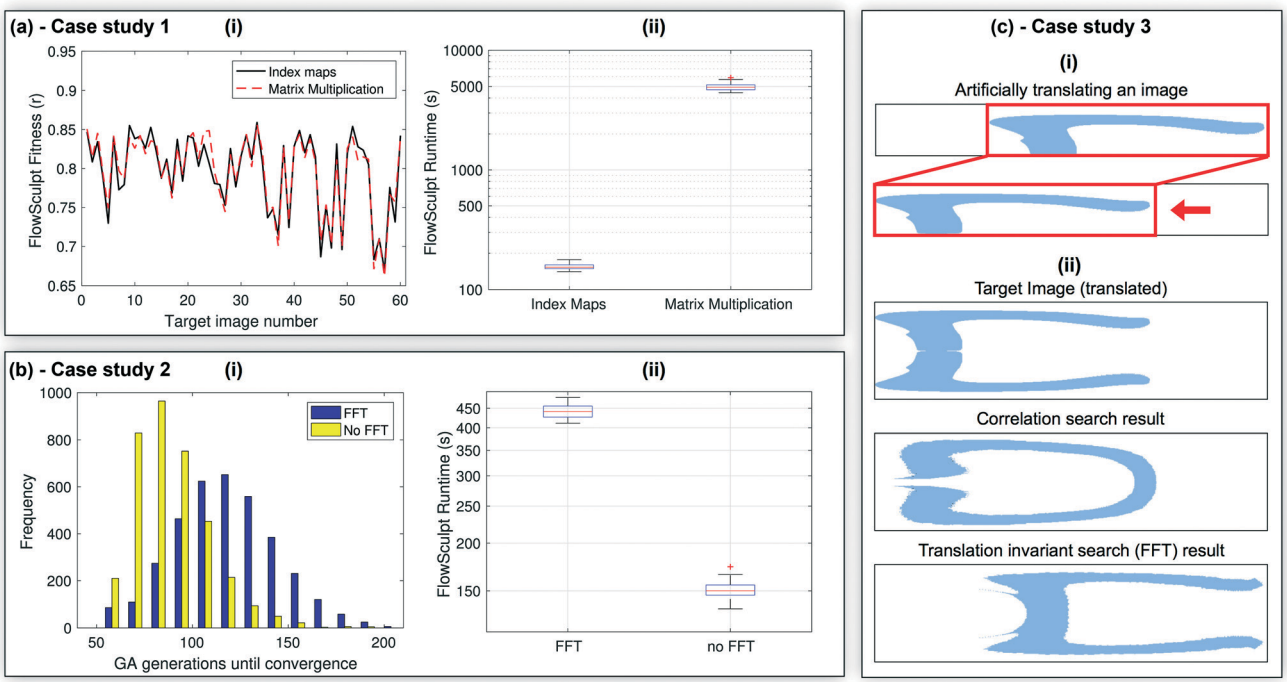

Fig. 7 (a) Results from case study 1, which compares the accuracy and performance of index maps to the previously used transition matrices, showing (i) matching values in GA fitness for the test set, with (ii) drastically reduced framework runtime. (b) Results from case study 2, a runtime comparison of FlowSculpt using the translation invariant (FFT) and pixel-to-pixel correlation (no FFT) fitness functions. (i) A translation invariant search space requires more generations within a GA to converge, which, in addition to increased computational effort in using the FFT, contributes to an increased runtime (ii) from an average of ≈145 seconds to ≈442 seconds. (c) Results from case study 3, comparing the pixel-to-pixel correlation fitness function to FFT-based translation invariance. (i) Target images with known solutions are artificially translated and (ii) used in FlowSculpt, with results shown from the correlation and translation invariant fitness functions.

campaign runtime exceeds FlowSculpts FFT-based translation invariant design, which searches a continuously translated space in an automated fashion.

5.1.3 Case study 3: translation invariant design. To test the efficacy of our translation invariant fitness function, we artificially translated each of the 60 target flow shapes from case study 1 as depicted in Fig. 7(c)(i), where a flow shape produced by the forward model is translated to the left edge of the channel cross-section. By translating flow shapes created by the forward model, we can test the FFT-based translation invariant fitness function against targets with known solutions. We simultaneously used the per–pixel correlation fitness function for comparison. In this case study, we allow FlowSculpt to design the inlet flow pattern.

The translation invariant search successfully found matching flow shapes for each altered target image, while the pixel-to-pixel correlation searches were pinned to the shifted location within the channel with poorly matching results. Fig. 7(c)(ii) shows an example target flow shape with a comparison of the result from the use of only the correlation function, and with use of the FFT-based translation invariant search. When FlowSculpt uses the correlation function only, it capably matches the horizontal location of the target flow shape, but fails to accurately capture details of the shape. On the other hand, the FFT-based translation invariant searches show a well-resolved flow shape, though at a different location within the channel cross-section.

Despite the considerably expanded search space, FlowSculpt was still able to effectively design both the inlet flow pattern and pillar sequences for known transformations. Going forward, users can use this fitness function to sculpt a particular flow shape without regard to its location in the channel, and avoid iterations of manually translating their target shape. On the other hand, for situations where the location of the sculpted flow is important, the default correlation function should be used. Still, these results are encouraging not only for the immediate use of translation invariant design, but for additional post-processing operations within FlowSculpt's fitness function that focus different user-specified qualities or quantities related to the sculpted flow.

5.2 Flow shape demonstration

FlowSculpt's expanded advection map library unlocks asymmetric flow shape design using half-pillars, and new flow physics from additional Reynolds numbers and channel aspect ratios. In Fig. 8, we show how FlowSculpt fares in arbitrary asymmetric shape design by searching for devices which create the capitalized letters of the Roman alphabet, and experimentally validate the device designs which create the letters in "UCLA". These target flow shapes were created in Microsoft's PowerPoint software (using the "Arial" font), exported as bitmap images, and loaded directly into FlowSculpt. For GA parameters, FlowSculpt was free to design the inlet flow pattern, all four Reynolds numbers

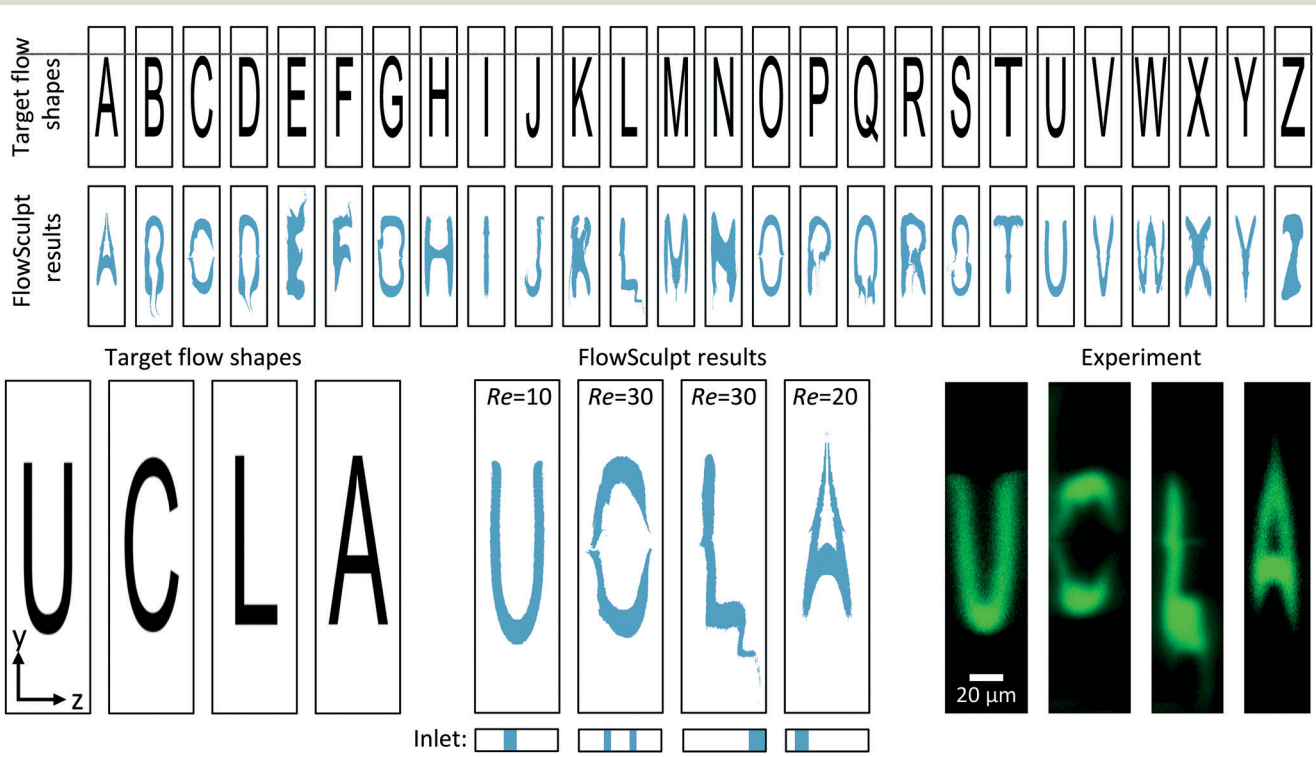

Fig. 8 Search results for all 26 (capitalized) letters in the Roman alphabet using h/w = 0.25 and Re = 10, 20, 30, 40, shown as rotated images for clarity. We show additional details for the letters in "UCLA", with each designs' Reynolds number shown inset with the image, and optimized inlet flow pattern below. These device designs were exported to OpenSCAD and fabricated using soft lithography (see Methods section), and sculpted flow was imaged using confocal microscopy.

were searched to maximize odds of success, and the correlation fitness function was used to ensure the letters are in similar region of the microchannel cross-section (rather than a scattered set of locations, a likely result in using the FFT search). Complete designs and per-pillar flow images for the "UCLA" devices at $h/w = 0.25$ in Fig. 8 are shown in ESI† Fig. S3–S6, with movies stitching each transformation together in sequence in ESI† SV01–SV06. Results for the higher aspect ratio channels (h/w = 0.5, 1.0) are in ESI† Fig. S8. We attribute differences between the experimental "UCLA" flow images and those predicted by FlowSculpt to errors in fabrication, which we verify using measured pillar diameters with the uFlow software's interpolative model[19] (see ESI† Fig. S7), and diffusion of the fluorescent dye as it traverses the length of the microchannel.

Close examination of the per-pillar image sequences (ESI† Fig. S3–S6) reveals a significant source of difficulty in solving the inverse problem in flow sculpting, particularly in the case of the "A" shape (ESI† Fig. S6): until the flow is deformed by the final pillar, it is not necessarily obvious what the resulting flow shape will look like. In other words, one cannot assume that pursuing the desired flow shape with every additional pillar will achieve a good result, which impedes the simple application of greedy algorithms. For the higher aspect ratio channels (ESI† Fig. S8), the modified flow physics appear less capable overall in matching the target alphabet flow shapes. This is not surprising given the previous observation that high aspect ratio channels produce less powerful flow deformations in the center of the channel, which is where the alphabet target flow shapes are placed.

We demonstrate multi-material and asymmetric design using 2- and 3-material target flow shapes, shown in Fig. 9. The left column of Fig. 9 shows two 3-material target shapes, each with two materials fully encapsulating a third material. These shapes are well represented by the top results found by FlowSculpt, with matching flow shapes shown in the right column. Asymmetric design is also shown with the last 4 target flow shapes, with 2-material designs having their material interface rotated at angles of 30°, 45°, 75°, and 90°. In the right column, FlowSculpt's top results competently match the targeted material partitions, including the rotated interface. We also performed an experiment with multi-material flow sculpting using two different materials, poly(ethylene glycol) diacrylate (PEGDA; $M_n \sim 575$; 437441, Sigma-Aldrich) and poly(propylene glycol) diacrylate (PPGDA; $M_n \sim 800$; 445024, Sigma-Aldrich), blending each material with ethanol (60% PEGDA, 40% ethanol; 90% PPGDA, 10% ethanol) to match their densities to avoid issues with buoyancy (see ESI† Fig. S9). However, their viscosities (PEGDA/ethanol: 6.99 mPa s; PPGDA/ethanol: 52.32 mPa s) remain unmatched. A 9-pillar multi-material flow sculpting device was designed using FlowSculpt (ESI† Fig. S9(b)), targeting a nested 2-material flow shape. We used a modified version of our in-house microparticle fabrication technique, transient liquid molding[5] (TLM, ESI† Fig. S9(a)), to create 3D microparticles in a 1200 μm × 300 μm channel (ESI† Fig. S9(c)). This experiment, along with previously published work,[5] shows that single-phase simulations using FlowSculpt's forward model are valid for some mismatch in fluid properties (viscosity, in this case), but results will vary depending on how large the disparity and which properties remain matched.

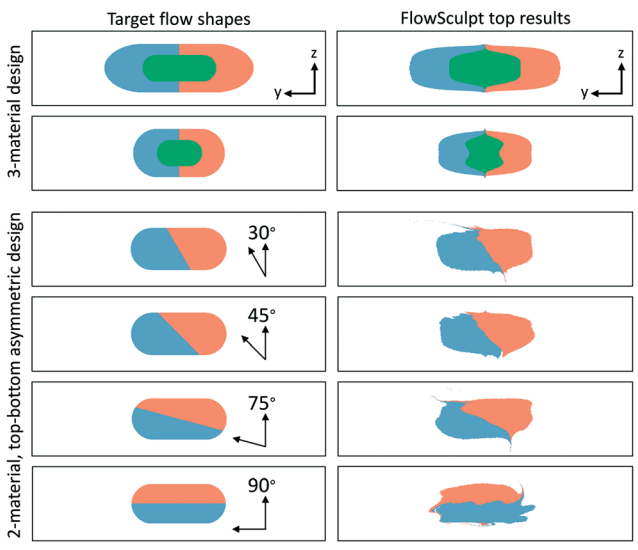

Fig. 9 Left column: Target flow shapes used to demonstrate multi-material and/or top–bottom asymmetric design. Right column: Top results from the FlowSculpt software for the left-column targets.

While these examples generally demonstrate a good match between their targets and results, iterative design is often required for arbitrary searches where there is no guarantee of finding a matching flow shape. This iterative process would begin with a user-provided target flow shape to serve as an initial guess. FlowSculpt will attempt to match this shape, but if the result is not sufficient for the user's application, this initial domain response can be used as an indication of what is possible, and inform subsequent FlowSculpt searches. This indicates the need for a first-pass "reality check" on target flow shapes – a problem beyond the scope of the FlowSculpt software in its current form, and likely made more difficult as the search space expands with new advection maps and choice of flow materials.

6 Conclusion

The FlowSculpt software provides an intuitive and easy to use graphical user interface for researchers to leverage when designing fluid deformations. It allows users to design a target flow shape graphically or to import an externally created image (*e.g.*, in Adobe Photoshop or PowerPoint), and gives a rich set of options for optimization including the number of threads to use for parallel evaluation, use of symmetry-breaking half-pillars, constraints on the size of the pillar sequence, and translation invariant search. The GUI is freely available as a user-friendly, cross-platform app, which we hope sees widespread adoption by the microfluidics community. We believe FlowSculpt sets a good standard for using software to relieve experts and non-experts alike of heavy

computational effort, and will help usher in a new generation of microfluidic devices for flow engineering, whether to advance microscale manufacturing processes, enhance existing bio-sensing platforms, or devise new bioengineering technologies. FlowSculpt's cross-platform architecture, combined with its newfound computational efficiency, should allow users with extremely modest computing hardware to engage in flow sculpting design optimization.

With this work, we not only present a highly functional application for design, but a significant step forward in its evolution. The elements we have added to the FlowSculpt framework vastly increase its capability through multi-material design, additional microfluidic components (half-pillars), and advanced fitness functions to modify the design space. Future work will likely continue along these routes, incorporating new geometries such as curved channels for Dean flow induced flow deformation, and more specialized fitness functions for unique flow shapes. We anticipate further development on user-provided microfluidic parts, allowing easier integration of FlowSculpt with researchers' existing microfluidics-based projects. Although there is some demonstration of multi-material flow sculpting using co-flowing fluids with varying physical properties in literature and in this work, additional characterization of multi-material constraints is warranted, as it is not clear how much the properties can differ before the forward model breaks down, or if a new model is required to more accurately capture interfacial physics; however, such a study is beyond the scope of this work, where we find the ability to design similar-property, multi-material flow shapes to be a highly useful advancement alone.

Eventually, the uFlow and FlowSculpt may merge into a single software, but their current architectures have different intent from the ground-up: uFlow for visualization (with little regard for efficiency), and FlowSculpt for optimization (which would be encumbered by uFlow's processes), making integration complex and a low-priority goal. Beyond these developments in the software, FlowSculpt itself can be brought into other regimes of hierarchical design, such as tailored 3D particles *via* stop-flow lithography,[4,6] transient liquid molding,[5] or continuous-flow lithography,[14] enabling a new class of complex microfluidic engineering. We also anticipate that FlowSculpt will find more general use in engineering flow streams as more disciplines become aware of its capability, especially in manipulating material concentrations or distributions in microfluidic flow applications.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

The authors thank Jesse Trujillo for his assistance in creating the advection maps used in this work. This research is supported in part by the National Science Foundation (NSF) through NSF-1306866 and NSF-1149365. We acknowledge the use of the Integrated Systems Nanofabrication Cleanroom at the California NanoSystems Institute (CNSI) at UCLA. Confocal laser scanning microscopy was performed at the CNSI Advanced Light Microscopy Shared Resource Facility at UCLA.

References

1 H. Amini, E. Sollier, M. Masaeli, Y. Xie, B. Ganapathysubramanian, H. A. Stone and D. Di Carlo, *Nat. Commun.*, 2013, **4**, 1826.
2 E. Sollier, H. Amini, D. E. Go, P. a. Sandoz, K. Owsley and D. Di Carlo, *Microfluid. Nanofluid.*, 2015, **19**, 53–65.
3 J. K. Nunes, C. Y. Wu, H. Amini, K. Owsley, D. Di Carlo and H. A. Stone, *Adv. Mater.*, 2014, **26**, 3712–3717.
4 K. S. Paulsen, D. Di Carlo and A. J. Chung, *Nat. Commun.*, 2015, **6**, 6976.
5 C. Y. Wu, K. Owsley and D. Di Carlo, *Adv. Mater.*, 2015, **27**, 7970–7978.
6 K. S. Paulsen, Y. Deng and A. J. Chung, *Adv. Sci.*, 2018, **5**, 1800252.
7 A. Choi, K. D. Seo, D. W. Kim, B. C. Kim and D. S. Kim, *Lab Chip*, 2017, **17**, 591–613.
8 G. M. Walker, J. Sai, A. Richmond, M. Stremler, C. Y. Chun and J. P. Wikswo, *Lab Chip*, 2005, **5**, 611–618.
9 Y. Zhang, C. Li, Y. Wu, Y. Zhang, Z. Zhou and B. Cao, *Biotechnol. Bioeng.*, 2019, **116**, 54–64.
10 N. Kimura, M. Maeki, Y. Sato, Y. Note, A. Ishida, H. Tani, H. Harashima and M. Tokeshi, *ACS Omega*, 2018, **3**, 5044–5051.
11 M. Selmi, M. H. Gazzah and H. Belmabrouk, *Sci. Rep.*, 2017, **7**, 1–11.
12 J. Xu, D. H. C. Wong, J. D. Byrne, K. Chen, C. Bowerman and J. M. DeSimone, *Angew. Chem., Int. Ed.*, 2013, **52**, 6580–6589.
13 K. J. McHugh, T. D. Nguyen, A. R. Linehan, D. Yang, A. M. Behrens, S. Rose, Z. L. Tochka, S. Y. Tzeng, J. J. Norman, A. C. Anselmo, X. Xu, S. Tomasic, M. A. Taylor, J. Lu, R. Guarecuco, R. Langer and A. Jaklenec, *Science*, 2017, **357**, 1138–1142.
14 L. A. Shaw, S. Chizari, M. Shusteff, H. Naghsh-Nilchi, D. Di Carlo and J. B. Hopkins, *Opt. Express*, 2018, **26**, 13543–13548.
15 R. Yuan, M. B. Nagarajan, J. Lee, J. Voldman, P. S. Doyle and Y. Fink, *Small*, 2018, **1803585**, 1803585.
16 C.-Y. Wu, D. Stoecklein, A. Kommajosula, J. Lin, K. Owsley, B. Ganapathysubramanian and D. Di Carlo, *Microsyst. Nanoeng.*, 2018, **4**, 21.
17 D. Stoecklein and D. Di Carlo, *Anal. Chem.*, 2019, **91**, 296–314.
18 D. Stoecklein, C.-Y. Wu, K. Owsley, Y. Xie, D. Di Carlo and B. Ganapathysubramanian, *Lab Chip*, 2014, **14**, 4197–4204.
19 D. Stoecklein, K. Owsley, C.-Y. Wu, D. Di Carlo and B. Ganapathysubramanian, *Microfluid. Nanofluid.*, 2018, **22**, 74.
20 S. Even and O. Goldreich, *Journal of Algorithms*, 1981, **2**, 311–313.
21 M. B. Giles and N. A. Pierce, *Flow, Turbul. Combust.*, 2000, **65**, 393–415.
22 B. Mohammadi and O. Pironneau, *Annu. Rev. Fluid Mech.*, 2004, **36**, 255–279.

23 S. D. Müller, I. Mezić, J. H. Walther and P. Koumoutsakos, *Comput. Fluids*, 2004, **33**, 521–531.
24 B. Ivorra, D. E. Hertzog, B. Mohammadi and J. G. Santiago, *Int. J. Numer. Methods Eng.*, 2006, **66**, 319–333.
25 D. R. Mott, P. B. Howell, K. S. Obenschain and E. S. Oran, *Mech. Res. Commun.*, 2009, **36**, 104–109.
26 C. A. Cortes-Quiroz, M. Zangeneh and A. Goto, *Microfluid. Nanofluid.*, 2009, **7**, 29–43.
27 D. Stoecklein, C.-Y. Wu, D. Kim, D. Di Carlo and B. Ganapathysubramanian, *Phys. Fluids*, 2016, **28**, 1–21.
28 D. Stoecklein, M. Davies, N. Wubshet, J. Le and B. Ganapathysubramanian, *J. Fluids Eng.*, 2017, **139**, 1–11.
29 D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
30 K. S. Paulsen and A. J. Chung, *Lab Chip*, 2016, **16**, 2987–2995.
31 C. Geuzaine and J. F. Remacle, *Int. J. Numer. Methods Eng.*, 2009, **79**, 1309–1331.
32 R. Yuster and U. Zwick, *ACM Transactions on Algorithms*, 2005, **1**, 2–13.
33 T. H. Cormen, C. Stein, R. L. Rivest and C. E. Leiserson, *Introduction to Algorithms*, McGraw-Hill Higher Education, 2nd edn, 2001.
34 M. R. Bailey, A. M. Pentecost, A. Selimovic, R. S. Martin and Z. D. Schultz, *Anal. Chem.*, 2015, **87**, 4347–4355.
35 M. Selmi, F. Echouchene, M. H. Gazzah and H. Belmabrouk, *IEEE Sens. J.*, 2015, **15**, 7321–7328.
36 V. Ribeiro, P. Coelho, F. Pinho and M. Alves, *Chem. Eng. Sci.*, 2012, **84**, 155–169.
37 G. Bradski, *Dr. Dobb's Journal of Software Tools*, 2000, **25**, 120–125, https://ci.nii.ac.jp/naid/10028167478/en/.
38 B. Srinivasa Reddy and B. N. Chatterji, *IEEE Trans. Image Process.*, 1996, **5**, 1266–1271.
39 M. Frigo and S. G. Johnson, *Proc. IEEE*, 2005, **93**, 216–231.