# MOPSA: A microfluidics-optimized particle simulation algorithm

Junchao Wang, Victor G. J. Rodgers, Philip Brisk, and William H. Grover

Citation: Biomicrofluidics **11**, 034121 (2017); doi: 10.1063/1.4989860
View online: http://dx.doi.org/10.1063/1.4989860
View Table of Contents: http://aip.scitation.org/toc/bmf/11/3
Published by the American Institute of Physics

# MOPSA: A microfluidics-optimized particle simulation algorithm

Junchao Wang,[1] Victor G. J. Rodgers,[1] Philip Brisk,[2] and William H. Grover[1,a)]
[1]*Department of Bioengineering, University of California, Riverside, California 92521, USA*
[2]*Department of Computer Science and Engineering, University of California, Riverside, California 92521, USA*

(Received 11 January 2017; accepted 12 June 2017; published online 26 June 2017)

Computer simulation plays a growing role in the design of microfluidic chips. However, the particle tracers in some existing commercial computational fluid dynamics software are not well suited for accurately simulating the trajectories of particles such as cells, microbeads, and droplets in microfluidic systems. To address this issue, we present a microfluidics-optimized particle simulation algorithm (MOPSA) that simulates the trajectories of cells, droplets, and other particles in microfluidic chips with more lifelike results than particle tracers in existing commercial software. When calculating the velocity of a particle, MOPSA treats the particle as a two-dimensional rigid circular object instead of a single point. MOPSA also checks for unrealistic interactions between particles and channel walls and applies an empirical correcting function to eliminate these errors. To validate the performance of MOPSA, we used it to simulate a variety of important features of microfluidic devices like channel intersections and deterministic lateral displacement (DLD) particle sorter chips. MOPSA successfully predicted that different particle sizes will have different trajectories in six published DLD experiments from three research groups; these DLD chips were used to sort a variety of different cells, particles, and droplets. While some of these particles are not actually rigid or spherical, MOPSA's approximation of these particles as rigid spheres nonetheless resulted in lifelike simulations of the behaviors of these particles (at least for the particle sizes and types shown here). In contrast, existing commercial software failed to replicate these experiments. Finally, to demonstrate that MOPSA can be extended to simulate other properties of particles, we added support for simulating particle density to MOPSA and then used MOPSA to simulate the operation of a microfluidic chip capable of sorting cells by their density. By enabling researchers to accurately simulate the behavior of some types of particles in microfluidic chips before fabricating the chips, MOPSA should accelerate the development of new microfluidic devices for important applications. *Published by AIP Publishing.* [http://dx.doi.org/10.1063/1.4989860]

## I. INTRODUCTION

The field of lab-on-a-chip and micro total analysis devices has been growing rapidly for nearly 40 years,[1] and microfluidic chips have found important applications in biological and chemical analyses. Many of these chips contain particles such as cells, microbeads, or droplets. For example, microfluidic cell sorters are emerging as powerful tools for point-of-care testing and biological research.[2,3]

As the variety of microfluidic devices increases, computer-based simulations of microfluidics have become more important.[4–6] Microfluidics researchers are increasingly using finite element analysis (FEA) software such as COMSOL Multiphysics (COMSOL Inc., Burlington,

a)Electronic mail: wgrover@engr.ucr.edu; URL: http://groverlab.org

MA), Autodesk CFD (Autodesk Inc., Venice, CA), and Fluent (ANSYS Inc., Canonsburg, PA) to simulate and optimize their microfluidic chip designs. These software tools excel at calculating the fluid velocity field in a chip. However, these tools are not well suited for simulating the behavior of particles like cells, beads, and droplets flowing in microfluidic chips. For example, Fig. 1 shows the results from using COMSOL Multiphysics to simulate the behavior of a deterministic lateral displacement (DLD) cell sorter. DLD sorters use arrays of micron-scale posts to sort cells based on their size. As cells flow through the chip and interact with the posts, large and small cells follow different paths and are therefore separated by the chip.[7,8] However, in the simulation results from COMSOL Multiphysics in Fig. 1, the larger 10 $\mu$m cells (red) and smaller 3 $\mu$m cells (green) follow exactly the same trajectory. If the simulation were accurate, the cells would have followed different trajectories. Additionally, by zooming in to the simulation results in Fig. 1(a), we see that the 10 $\mu$m cell actually overlaps with the wall of a triangular post (an impossible situation). These observations confirm that the particle tracing algorithm in COMSOL Multiphysics treats the particle as a mathematical point with no area or volume, an assumption that can be acceptable in macro-scale physics but not in microfluidics. This assumption leads to additional problems: if the simulated single-point particle in COMSOL Multiphysics is located on a mesh node with a fluid velocity magnitude of zero, the particle will remain stuck at that location forever [Fig. 1(b)]. This sticking behavior remains regardless of which boundary condition is chosen for the wall in COMSOL Multiphysics (*freeze, stick, bounce, disappear*, or *reflect diffusely*).

The lack of commercial software tools that accurately simulate particles in microfluidics has profound consequences. For example, researchers developing new cell sorters like the DLD cell sorter shown in Fig. 1 typically have to fabricate and test these chips to determine whether they sort the desired cells. While some empirical models for predicting the sorting behavior of DLD chips do exist for chips with cylindrical posts,[9] these models do not easily extend to posts with arbitrary shapes like the triangles shown in Fig. 1. And with each chip design iteration requiring fabrication and testing in the lab, it can take months for researchers to develop a functional device. This slows the progress of research and keeps valuable research tools and lifesaving medical diagnostics out of the hands of the people who need them.

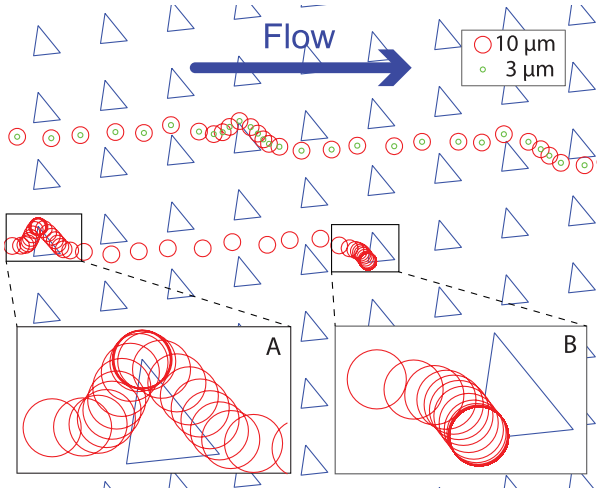


FIG. 1. Results from using the commercial finite element analysis software COMSOL Multiphysics to simulate the behavior of two sizes of cells flowing through a deterministic lateral displacement (DLD) microfluidic cell sorter chip containing an array of triangular posts. While this DLD chip was designed to separate cells based on size, in this simulation the small cell (3 $\mu$m diameter; green) and large cell (10 $\mu$m; red) follow exactly the same trajectory and are not separated. Additionally, the simulated cells sometimes impossibly overlap the chip's triangular posts (a) or stick permanently (b). Deficiencies like these make it difficult to accurately simulate flowing cells and other particles in microfluidics using existing commercial software tools.

To address this problem, we developed a microfluidics-optimized particle simulation algorithm (MOPSA) that is capable of simulating lifelike trajectories for some particles in situations where existing software tools fail. MOPSA treats a particle as a 2D rigid circular object instead of a single point when calculating the particle's velocity. MOPSA also checks whether the particle overlaps a solid object (a post or channel wall) at each simulated time step. If overlap is detected, MOPSA applies an empirical correcting function, *wallEffect*, to shift the particle to a new position where it will not overlap walls. By providing researchers with accurate simulations of some types of particles in microfluidic devices, MOPSA should accelerate the development of new microfluidic devices for important applications in biological research, medical diagnostics, and beyond.

Assuming that a particle is a rigid circle means that MOPSA cannot directly simulate the deformability or non-spherical shapes of some particles. In spite of this limitation, by assuming that the particles are rigid circles, MOPSA nonetheless predicts particle trajectories that are consistent with the experimentally observed behavior of published DLD chips sorting some types of deformable and non-spherical cells and droplets. However, undoubtedly there are particles that cannot be approximated as rigid circles and simulated successfully using MOPSA; for these particles other simulation approaches may be necessary and could even be incorporated into future versions of MOPSA (as discussed in the Conclusions).

## II. SIMULATION

Simulating a microfluidic chip using MOPSA requires two steps:

1. Calculate the fluid velocity field in the chip using conventional finite element analysis software
2. Calculate the trajectory followed by a particle through the chip using MOPSA

In this proof-of-concept demonstration, we used COMSOL Multiphysics to calculate the fluid velocity field and MATLAB to implement MOPSA. We used this approach to simulate three different microfluidic chip models:

1. The *cross channel model* is a typical microfluidic cross-shaped intersection with 200 $\mu$m channel widths. Junctions like this are extremely common in microfluidic chips,[10–13] so assessing the performance of MOPSA in these intersections is very important.
2. The *triangular DLD model* was chosen because of its complexity. Arrays of hundreds of triangular posts offer many opportunities for interactions between particles and the posts and challenge the robustness of MOPSA.
3. The *cylindrical DLD model* enables comparisons between MOPSA and published experimental results. The vast majority of DLD chips in the literature use arrays of cylindrical posts, and we use MOPSA to simulate the paths followed by particles in six published DLD experiments by three different research groups.[7,14,15]

While we chose these three models for this proof-of-concept demonstration, we note that, in principle, MOPSA can simulate virtually any microfluidic chip design containing flowing particles, as long as the chip can be modeled in two dimensions (MOPSA is currently a two-dimensional simulation, although it could easily be extended to three dimensions as discussed in the Conclusions) and as long as the particles can be approximated as rigid circles or spheres. Finally, to compare MOPSA to existing commercial software tools for particle simulation in microfluidics, we also simulated these chip models entirely in COMSOL Multiphysics (without using MOPSA).

### A. Calculating fluid velocity field in COMSOL

For each microfluidic chip model simulated here, the fluid velocity fields were calculated using finite element analysis software (COMSOL Multiphysics). For accuracy, we used the *Laminar Flow* physics module and a customized free triangle mesh with a maximum mesh size that was equal to or less than the diameter of the smallest particle that would be simulated using that mesh. In the *cross channel model*, the maximum mesh size was 8 $\mu$m (for simulating a

particle diameter of 50 $\mu$m). Inlets were assigned an inlet boundary condition of 5 mm/s normal inflow velocity. In the *triangular DLD models*, the maximum mesh size was 1 $\mu$m (for simulating particle diameters of 1, 3, 10, and 12 $\mu$m). In the *cylindrical DLD models*, the maximum mesh size was a quarter of the difference between two simulated particles' diameters (e.g., for simulating a DLD chip separating 8 and 9 $\mu$m beads, the maximum mesh size was 0.25 $\mu$m). Inlets were assigned an inlet boundary condition of 1 mm/s normal inflow velocity. In all models, outlets were assigned an outlet boundary condition of 0 Pa pressure, the remaining boundaries were walls (no-slip boundary condition), and the material filling the channels was water under incompressible flow. A stationary solver was used for calculating the fluid velocity field.

ALGORITHM 1 Main MOPSA algorithm

**Require:** Velocity profile in $x$ and $y$ direction of model ($x.csv$, $y.csv$), diameter of particle ($D_\mathrm{p}$), initial position of particle ($position_\mathrm{i}$), model geometry file (geometry.dxf), resolution of generating streamlines ($resolution$), $vx_\mathrm{pre}$, $vy_\mathrm{pre}$, $\beta$.

**Ensure:** Simulated particle trajectory ($trajectory$)

1: $polylines \leftarrow geometry.dxf$ ▷ Load microfluidic chip geometry from DXF file.

2: $x \leftarrow x.csv$ ▷ Load pre-simulated fluid velocity profiles.

3: $y \leftarrow y.csv$

4: $node \leftarrow x.csv$ ▷ Load mesh information from x.csv.

5: $node_\mathrm{sorted} \leftarrow \textsc{sortrows}(node)$ ▷ Sort nodes based on x-direction.

6: $geometry_{boundary}[x_{min}\, x_{max}\, y_{min}\, y_{max}] \leftarrow [\textsc{min}(node(:,1)) \quad \textsc{max}(node(:,1)) \quad \textsc{min}(node(:,2) \quad \textsc{max}(node(:,2))]$ ▷ Define region of interest.

7: $position_c \leftarrow position_i$ ▷ Initialize current position of particle.

8: $trajectory \leftarrow position_i$ ▷ Initialize trajectory of particle.

9: **while** $position_c \in geometry_{boundary}$ **do** ▷ Move to next time step if particle is still in region of interest.

10: $nodes_{covered} \leftarrow \textsc{findCoveredNodes}(D_p, position_c, node_{sorted}, node)$ ▷ Identify mesh nodes covered by particle.

11: $v_x \leftarrow \textsc{getMean}(nodes_{covered}, x)$ ▷ Set particle velocity to average fluid velocity of covered mesh nodes.

12: $v_y \leftarrow \textsc{getMean}(nodes_{covered}, y)$

13: $v_y \leftarrow v_y \cdot \beta$ ▷ Scale lateral ($y$) component of particle velocity by $\beta$ if necessary.

14: **if** $v_\mathrm{x} = 0$ and $v_\mathrm{y} = 0$ **then** ▷ If particle has zero velocity (is stuck)…

15: $v_x \leftarrow vx_{pre}$ ▷…use the particle's velocity from the previous time step instead.

16: $v_y \leftarrow vy_{pre}$

17: **end if**

18: $position_n \leftarrow position_c + resolution \cdot [v_x\, v_y]$ ▷ Calculate position of particle at next time step.

19: $position_n \leftarrow \textsc{wallEffect}(D_p/2, position_n, polylines)$ ▷ Apply wallEffect (see Algorithm 2).

20: $trajectory \leftarrow [trajectory; position_n]$

21: $position_c \leftarrow position_n$ ▷ Record new particle position.

22: $vx_{pre} \leftarrow v_x$ ▷ Record new "previous" particle velocity for use in next iteration.

23: $vx_{pre} \leftarrow v_y$

24: **end while**

### B. Simulating particle trajectory with MOPSA

MOPSA is summarized in pseudocode in Algorithm 1. We implemented MOPSA in MATLAB to leverage MATLAB's strengths in matrix calculation. Using the built-in MATLAB function *INPOLYGON* significantly simplifies the MOPSA code. In principle, the pseudocode shown here could be implemented in any programming language.

MOPSA first imports the geometry of the microfluidic chip (the design of the chip in DXF format[16]), the fluid velocity field calculated by COMSOL Multiphysics, and the description of the mesh used by COMSOL (lines 1–4 of Algorithm 1). Instead of treating the particle as a single mathematical point, MOPSA uses the average velocity of all of the mesh nodes covered by the particle to determine the particle's velocity profile (lines 11–12). $v_x$ represents the magnitude of the velocity vector in the $x$ direction, and $v_y$ represents the velocity in the $y$ direction.

As shown in Algorithm 1 line 13, MOPSA uses an empirically obtained parameter $\beta$ to "weight" the components of the particle velocity vector in direction orthogonal to fluid flow (the $y$ direction). For the *cross channel model*, $\beta = 1$ (meaning that the velocities calculated by COMSOL are used as-is). For the *triangular* and *cylindrical DLD* models, $\beta = 1.45$. Details on how this value was determined are provided in Results and Discussion.

MOPSA next considers the possibility of particles with zero velocity. In theory, as long as there is fluid flow in a microfluidic channel, mesh nodes with zero fluid velocity should exist only at channel walls. However, in practice, a finite mesh size makes it possible for software like COMSOL Multiphysics to predict that nodes *near* channel walls may also have zero fluid velocity. A particle located at a mesh node with zero fluid velocity can become permanently trapped. To detect this situation, MOPSA utilizes a conditional statement (line 14 in Algorithm 1) to determine if the velocity of the particle in the time step will be zero. If so, MOPSA uses the particle's non-zero velocity from the *previous* time step for the current time step calculation (lines 15 and 16). As long as the amount of time between steps [*resolution*, defined in Eq. (1) below] is sufficiently small, this substitution does not seem to affect the accuracy of the simulation. In this manner, MOPSA ensures that the particle will not get permanently stuck. Alternatively, for applications in which particle sticking is desired (for example, using antibody-coated posts to capture cells of interest in a microfluidic chip[17]), a threshold can be added to allow particle sticking below a user-specified particle velocity.

MOPSA then uses Eq. (1) to calculate the particle's position at the next time step[18]

$$position_n = position_c + resolution \cdot [v_x\, v_y], \tag{1}$$

where $position_c$ ($\mu$m) is the current position of the particle, $position_n$ ($\mu$m) is the position of the particle in the next time step, *resolution* (s) is the amount of time between each calculation step, and $v_x$ and $v_y$ ($\mu$m/s) are the average fluid velocities of the mesh nodes covered by the particle, as defined earlier. The variable *resolution* is a parameter that can be customized by the user. A smaller *resolution* makes the simulation more accurate but also more time consuming. If the flow through a chip is relatively fast, then a small value for *resolution* is needed to accurately predict the path followed by a particle in the fluid. If the flow is relatively slow, then a larger *resolution* can be used without sacrificing simulation accuracy. To find an optimal value for *resolution*, a MOPSA simulation can be repeated with successively smaller values for *resolution*. Eventually the predicted particle trajectories will converge to a single trajectory that will not change with additional decreases in *resolution*; this is the optimal value for *resolution* for a given application.

Eq. (1) assumes that:

1. The density of the particle is the same as its surrounding fluid.
2. The net force applied to the particle is always zero.
3. Particles do not interfere with each other or channel walls.
4. Particles have smooth surfaces, so friction between fluid and particle surfaces is negligible.
5. The material of the particle is homogeneous.

If these assumptions are unsuitable for a particular application, MOPSA can be modified by replacing Eq. (1) with a more complicated drag law such as Stokes' Law[19] or the work of Haider and Levenspiel[20] (we demonstrate a modification of MOPSA later in this work).

ALGORITHM 2 function wallEffect

**Require:** microfluidic chip wall geometry information (*polylines*), radius of particle ($R_p$), current position of particle ($position_c$), particle mapping parameter ($k$)

**Ensure:** shifted position of particle ($position_s$)

1: **function** WALLEFFECT($R_p$, $position_c$, *polylines*)

2: $x_p \leftarrow R_p \cdot \cos(0 : (2\pi/k) : 2\pi) + position_c(1, 1)$ ▷ Calculate locations of the $k$ points representing the particle boundary.

$y_p \leftarrow R_p \cdot \sin(0 : (2\pi/k) : 2\pi) + position_c(1,2)$
**for** $i = 0 \rightarrow total_{polylines}$ **do**
  $polyline_c \leftarrow polylines[i]$ ▷ Obtain the next polyline representing a channel feature (post, wall, etc.)
  **if** $polyline_c(1,:) = polyline_c(end,:)$ **then** ▷ Confirm polyline is closed.
    $x_{polyline} \leftarrow polyline_c(:,1)$
    $y_{polyline} \leftarrow polyline_c(:,2)$
    $in \leftarrow$ INPOLYGON$(x_p, y_p, x_{polyline}, y_{polyline})$ ▷ Determine how many boundary points of the particle are inside the polyline.
    $count_{in} \leftarrow 0$
    $points_{in} \leftarrow []$
    **for** $j = 0 \rightarrow total_{in}$ **do**
      **if** $in(j) = 1$ **then**
        $count_{in}++$
        $points_{in} \leftarrow [points_{in}; x_p\ y_p]$
      **end if**
    **end for**
    **if** $count_{in} > k$ or $count_{in} = 0$ **then** ▷ If the particle *does not* overlap with the wall…
      $position_s \leftarrow position_c$ ▷…keep the current position of the particle.
    **else** ▷ However, if the particle *does* overlap with the wall:
      $point_{middle} \leftarrow$ [MEAN$(points_{in}(1,:), points_{in}(end,:))$]
      $direction \leftarrow$NORM$(position_c - point_{middle})$ ▷…calculate the shift direction as a normal vector…
      $position_s \leftarrow position_c + direction \cdot (count_{in}/k) \cdot R_p \cdot w$ ▷…and apply the shift to the particle location.
      BREAK ()
    **end if**
  **end if**
**end for**
**return** $position_s$ ▷ Return the new particle location to the main MOPSA algorithm.
**end function**

After the particle's position at the next time step has been calculated, MOPSA's *wallEffect* function (Algorithm 2) determines if the particle will collide or overlap with solid walls. As shown in Fig. 2, *wallEffect* uses a large number of points ($k = 200$) to map the boundary of the particle located at position $a$. If any of these $k$ points are located inside a solid wall (e.g., a triangular post, a channel wall, or any other device feature), then the algorithm recognizes that the particle is overlapping the wall and compensates by shifting the particle's position away from the wall to a new location $a'$. The direction of this shift is determined by first identifying the two points of contact between the particle perimeter and the wall ($b$ and $c$), then defining a line segment $bc$ between these two points, and finally computing the midpoint $d$ of the line segment. The direction of the particle shift is perpendicular to line $bc$ (roughly pointing in the opposite direction of the fluid flow) and is calculated using Eq. (2)

$$position_s = position_c + direction \cdot \frac{count_{in}}{k} \cdot R_p \cdot w \tag{2}$$

where $position_s$ is the shifted position of the particle, $position_c$ is the initial (wall-overlapping) position of the particle, *direction* is a normal vector describing the shifting direction, $k$ is the total number of points used to map the perimeter of the particle, $count_{in}$ is the number of those perimeter mapping points located inside the solid wall, $R_p$ is the radius of the particle, and $w$ is a weight factor that can be adjusted for different models ($w = 1$ was used in this work). Once this algorithm has corrected the overlapping position of the particle, MOPSA moves on to the next simulation time step.

The parameters *resolution* in Eq. (1) and $w$ in Eq. (2) provide a mechanism for fine-tuning the *wallEffect* algorithm. For example, if the algorithm predicts an unrealistically large shift distance, reducing *resolution* will decrease the amount of overlap between the particle and wall

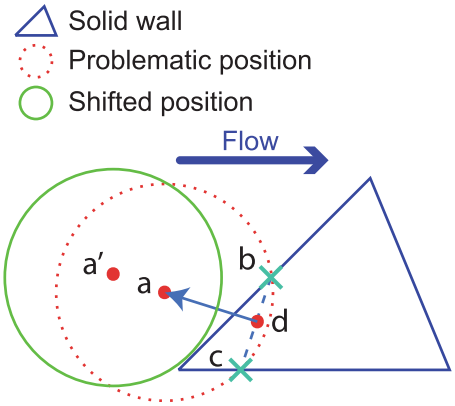


FIG. 2. Function of the *wallEffect* algorithm. A particle centered at point *a* (*position*$_c$ in Algorithm 2) has its boundary represented by 200 points (depicted as small red points). If any of these points overlap with a solid chip feature, such as the triangular post shown (blue), the algorithm shifts the particle to a new location *a*′ (green boundary; *position*$_s$ in Algorithm 2) in a direction perpendicular to the midpoint *d* of line *bc* (defined by the cyan ×-shaped points on the intersections between the particle boundary and the edges of the post). The distance of the shift is determined by Eq. (2). In this illustration, the area of overlap between the particle and post and the distance of the shift are intentionally exaggerated for clarity. In practice, choosing a properly small value of *resolution* [Eq. (1)] will ensure that the overlap and shift distance are small and the shift direction will be approximately perpendicular to the edge of the post.

(and thus decrease the amount of correction applied by the *wallEffect*. If the algorithm predicts an unrealistically small shift distance, increasing $w$ will increase the shift distance. Finally, if the geometry of a wall has concave features, *wallEffect* may yield unexpected results because the particle perimeter could intersect with the wall in more than two points. In this case, changing $w$ could compensate for unexpected results from concave wall geometries. In any case, $w = 1$ was used for all of the simulations presented here.

### C. Comparison with existing commercial particle tracers

To compare the performance of MOPSA to an existing commercial particle tracer, we repeated all of our particle simulations using the built-in *Particle Tracing for Fluid Flow* physics module in COMSOL Multiphysics. The simulations used a time-dependent solver, and a "drag force" boundary was added into the physics to use Stokes' Law for particle trajectory calculation. For the *triangular DLD model*, the inlet was assigned an inlet boundary condition of 100 particles per release with a uniform distribution. The particles were assigned diameters of 1, 3, 10, or 12 $\mu$m (although, as described above, these diameters do not seem to affect the simulation results in COMSOL Multiphysics). The outlet was assigned an outlet boundary condition of *freeze wall*. The six *cylindrical DLD model* chips we simulated used the same conditions as the *triangular DLD model* but with 10 particles per release and the particle diameters shown in Table I. We also tested a range of boundary conditions for the post walls in the DLD models, including *bounce, diffuse scattering*, and *general reflection*. Finally, for the *cross channel model*, the inlet was assigned an inlet boundary condition of 10 particles per release (50 $\mu$m particle diameter) with uniform distribution and the outlet was assigned an outlet boundary condition of *freeze wall*.

## III. RESULTS AND DISCUSSION

### A. Simulating the cross channel model

The fluid velocity field of the *cross channel model* was solved using COMSOL Multiphysics and is shown in Fig. 3(a). This model simulates a cross-channel with fluid entering from the left and exiting out the top, right, and bottom. For rigid spherical particles originating at many locations across the left-hand entrance channel, COMSOL's particle tracer and MOPSA predict very

TABLE I. Experimental details from six published deterministic lateral displacement (DLD) particle separations. These experiments are reproduced by MOPSA in Fig. 7. The parameters $\lambda$, $\Delta\lambda$, and $G$ describe the DLD device design and are defined in Fig. 6.

| Figure | $\lambda$ ($\mu$m) | $\Delta\lambda$ ($\mu$m) | G ($\mu$m) | Sorted particle types and sizes | Figure in reference |
|---|---|---|---|---|---|
| 7(a) | 80 | 6 | 25 | 8 $\mu$m and 9 $\mu$m beads | Fig. 2(a) in Li *et al.*[14] |
| 7(b) | 90 | 8 | 25 | 9 $\mu$m and 10 $\mu$m beads | Fig. 2(b) in Li *et al.*[14] |
| 7(c) | 80 | 2 | 20 | 2 $\mu$m platelets and 6 $\mu$m red blood cells | Fig. 4(a) in Li *et al.*[14] |
| 7(d) | 80 | 5 | 20 | 6 $\mu$m red blood cells and 10 $\mu$m white blood cells | Fig. 4(b) in Li *et al.*[14] |
| 7(e) | 8 | 0.8 | 1.6 | 0.4 $\mu$m and 1.0 $\mu$m beads | Fig. 2(a) in Huang *et al.*[7] |
| 7(f) | 120 | 6 | 60 | 11 $\mu$m and 30 $\mu$m droplets | Fig. 2(c) in Joensson *et al.*[15] |

similar particle trajectories. However, for particles that start close to a channel wall, MOPSA's simulation is more realistic than COMSOL's. Figure 3(b) shows trajectories predicted by COMSOL (red) and MOPSA (green) for a 50 $\mu$m diameter particle flowing adjacent to the channel wall. In the trajectory predicted by COMSOL, the particle overlaps with the channel wall as it goes around the corner. In contrast, MOPSA detects these channel walls and keeps the particle from overlapping them. The realistic wall interactions simulated by MOPSA result in a particle trajectory that is 18 $\mu$m (36% of the particle's size) farther to the right than the trajectory calculated by COMSOL. For simple microfluidic chips with a single intersection, the particle tracing errors introduced by software like COMSOL Multiphysics may be acceptable and may not affect the overall accuracy of the simulation. However, for microfludic chips with hundreds of intersections in parallel (like the hydrodynamic filter of Yamada *et al.*[10]), small errors in each intersection could combine to form a large net error associated with the path followed by particles flowing across the entire chip. In these cases, MOPSA should provide a much more accurate particle trajectory than existing software tools.

### B. Simulating deterministic lateral displacement (DLD)

We also tested MOPSA on the *triangular deterministic lateral displacement (DLD) model* cell sorter to determine how the algorithm performs in a chip with more complex geometry. Figure 4 shows the predicted particle trajectories for a 10 $\mu$m diameter rigid spherical particle flowing through an array of triangular posts, calculated using COMSOL Multiphysics (red) and

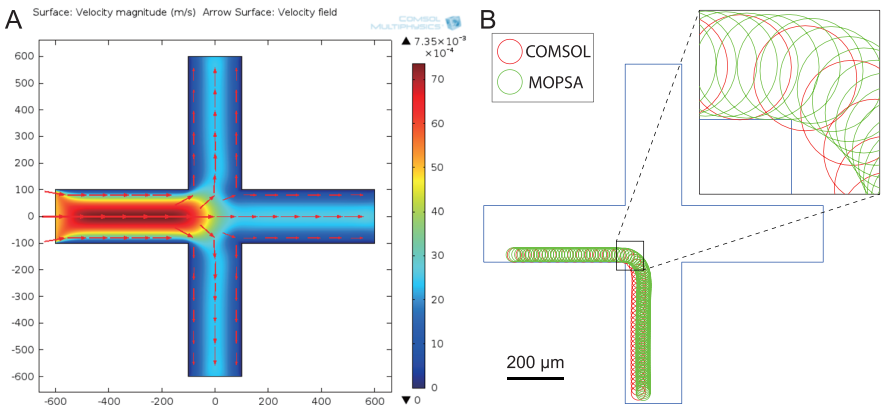


FIG. 3. (a) Fluid velocity field of the *cross channel model* obtained from COMSOL Multiphysics (dimensions in $\mu$m and velocity in m/s). (b) Simulated trajectories of a 50 $\mu$m diameter particle traveling through the *cross channel model*. In the trajectory calculated by the particle tracer in COMSOL Multiphysics (red outlines) the particle overlaps with the channel wall (an impossibility). However, in the trajectory calculated by MOPSA (green outlines) the particle remains separate from the channel wall and exits the intersection offset 18 $\mu$m from the COMSOL-predicted trajectory (a significant difference for a particle this size).

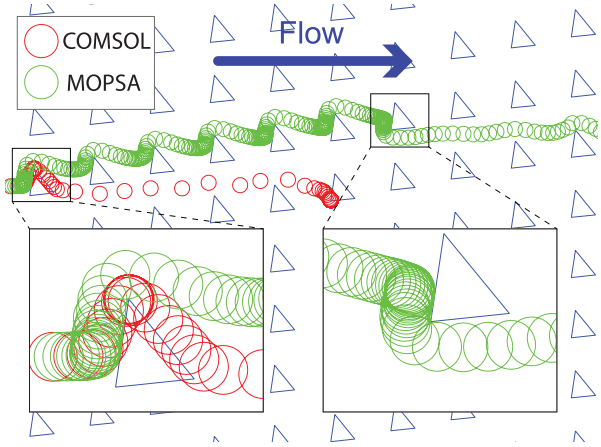


FIG. 4. Predicted trajectories for a 10 $\mu$m diameter particle traveling through an array of triangular posts, obtained using either COMSOL Multiphysics (red) or MOPSA (green). While the COMSOL trajectory allows the particle to overlap with solid posts and stick permanently, the MOPSA trajectory does not. Additionally, the trajectory predicted by MOPSA exhibits the lateral particle displacement expected in DLD chips like this one.

MOPSA (green). In both simulations, the particle starts at the same location. Since COMSOL treats the particle as a mathematical point, COMSOL's trajectory allows the particle to overlap with a triangular post, which is physically impossible. Additionally, later in the COMSOL simulation the particle encountered a zero-velocity mesh node and permanently stopped. In contrast, the trajectory calculated by MOPSA is much more consistent with the known mechanism of DLD. Repeated interactions with triangular posts laterally displace the particle, and no overlapping or sticking with posts is observed (even after repeating the simulation shown in Fig. 4 hundreds of times with different starting locations for the particle). The design of this chip in DXF format, fluid velocity field, and particle trajectory data are all available for download in online supplementary material.

The goal of DLD is to separate different cells and other particles based on their size: smaller particles follow the fluid streamlines straight through the device, while larger particles are "bumped" laterally by the posts and follow a diagonal path through the device. To confirm that MOPSA can simulate the particle sorting capabilities of a DLD chip, in Fig. 5 we used it to simulate the trajectories of 1, 10, and 12 $\mu$m diameter rigid spherical particles flowing

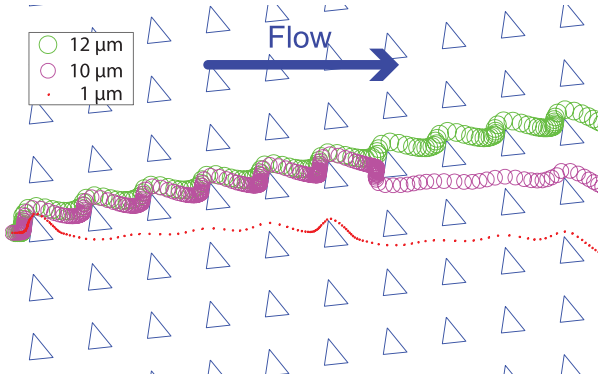


FIG. 5. Using MOPSA to predict the trajectories of 1 $\mu$m (red), 10 $\mu$m (purple), and 12 $\mu$m (green) diameter particles flowing through a DLD chip containing an array of triangular posts. The simulation predicts that the smallest particle will flow straight through the array and the largest particle will be displaced laterally by interactions with the posts. This is consistent with the known operation of DLD devices. The mid-sized particle (10 $\mu$m; purple) is initially displaced laterally but then follows a straight path; this suggests that this particle may be close to the critical diameter for this DLD chip design.

through an array of triangular posts. Although particles of all three sizes start in the same location, MOPSA predicts that the particles will exit the chip in different locations: the small 1 $\mu$m particle follows the fluid streamlines and moves straight through the chip, and the large 12 $\mu$m particle is "bumped" by each post and moves diagonally through the chip. The mid-sized 10 $\mu$m particle initially follows a diagonal path but then switches to a straight-through path, leading to a trajectory that lies between the other two particles' trajectories. This suggests that the critical diameter for this DLD chip design (the size of the smallest particle that is laterally displaced in the chip) is close to 10 $\mu$m. In contrast, particles of different sizes follow exactly the same trajectory in COMSOL's particle tracer (Fig. 1). These results confirm that MOPSA can successfully simulate size-based DLD separations that are difficult or impossible to simulate using existing commercial software, at least for rigid spherical particles. The design of this chip in DXF format, fluid velocity field, and particle trajectory data are all available for download in online supplementary material.

### C. Simulating published DLD experiments

We next used MOPSA to predict the paths followed by different sized particles in several published DLD chips. We identified six experiments in three published papers[7,14,15] in which the chip designs and experimental conditions are described in sufficient detail to allow us to simulate them using MOPSA. These chips use arrays of cylindrical posts with dimensions defined in Fig. 6 and summarized in Table I.

We first simulated the DLD chip described by Li *et al.*[14] who used this chip to sort 8 $\mu$m and 9 $\mu$m diameter rigid spherical beads. We found that with a value of $\beta = 1.45$, the MOPSA predicts that the 8 $\mu$m and 9 $\mu$m beads will follow different paths through the DLD chip [Fig. 7(a)]; this is consistent with the observed bead sorting behavior of this DLD chip.[14]

We then left the value of $\beta = 1.45$ unchanged as we used MOPSA to predict the paths followed by particles through five additional published DLD chips from three research groups [Figs. 7(b)–7(f)]. In their original publications,[7,14,15] these chips were used to sort a wide variety of particle sizes (from 0.4 $\mu$m to 30 $\mu$m diameters) and types (beads, droplets, red blood cells,

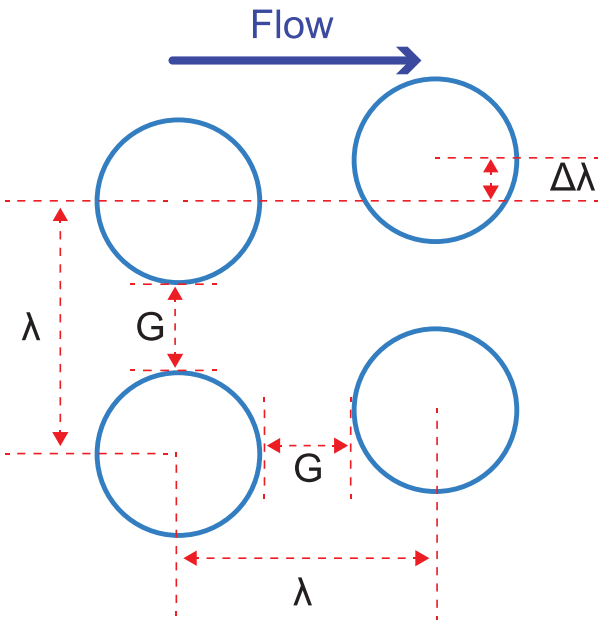


FIG. 6. Illustration of the key dimensions in a traditional cylindrical-post-based DLD chip: spacing between posts ($\lambda$), gap between posts ($G$), and offset between rows of posts ($\Delta\lambda$).

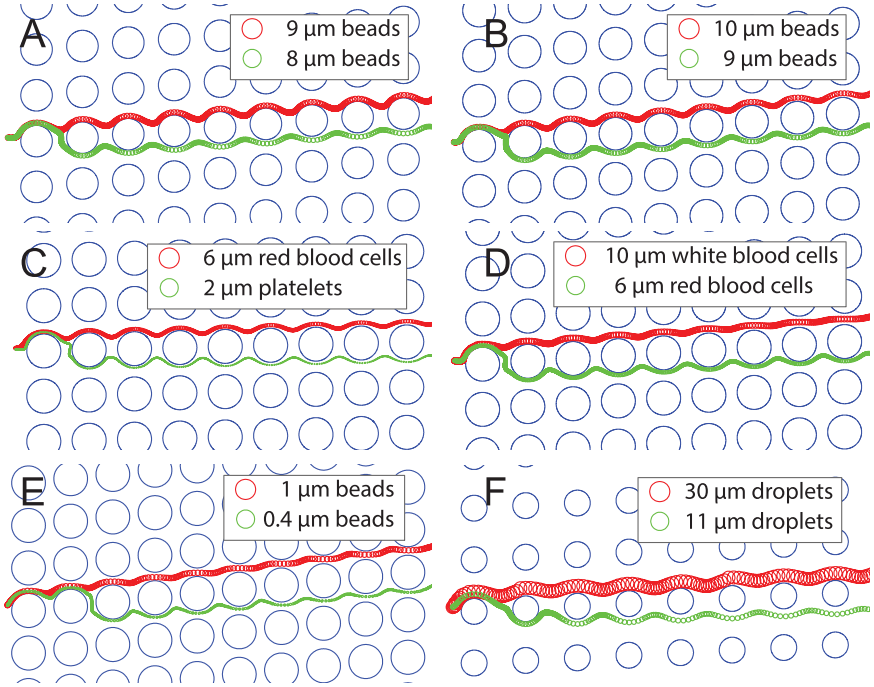


FIG. 7. Using MOPSA to predict particle trajectories in six published deterministic lateral displacement (DLD) experiments from three research groups.[7,14,15] These simulations are based on (a) a chip that separates 8 $\mu$m and 9 $\mu$m beads,[14] (b) a chip that separates 9 $\mu$m and 10 $\mu$m beads,[14] (c) a chip that separates 2 $\mu$m platelets and 6 $\mu$m red blood cells,[14] (d) a chip that separates 6 $\mu$m red blood cells and 10 $\mu$m white blood cells,[14] (e) a chip that separates 0.4 $\mu$m and 1.0 $\mu$m beads,[7] and (f) a chip that separates 11 $\mu$m and 30 $\mu$m droplets.[15] In each case, MOPSA's prediction that the different-sized particles follow different paths is consistent with the experimentally observed particle sorting behavior of these DLD chips.[7,14,15]

white blood cells, and platelets). Some of these particles are not rigid or spherical—droplets are not rigid, and red blood cells, white blood cells, and platelets are neither rigid nor spherical—so one may rightly expect that MOPSA (which assumes rigid spherical particles) may not be able to accurately simulate the trajectories of these particles. However, MOPSA nonetheless successfully predicted that the different-sized particles follow different paths through the chip. This prediction is consistent with the particle-sorting behavior reported by the creators of these chips. In contrast, the built-in particle tracer in COMSOL Multiphysics predicted no differences in the paths followed by the different particles (see supplementary material Figs. 2–7). Undoubtedly there are some particles whose deformability and non-spherical shapes will render MOPSA unable to accurately simulate the behavior of these particles, but at least for the particle types and sizes we studied, MOPSA's assumption of rigid and spherical particles did not seem to adversely affect the quality of the simulation results. The chip design DXF files, fluid velocity profiles, and predicted particle trajectories from both MOPSA and COMSOL for each of the six simulations in Fig. 7 are available in supplementary material.

In some of the DLD chip simulations in Fig. 7, MOPSA predicted that the smaller particles will follow a somewhat jagged path when the particles flow vertically between two rows of posts. For example, in Fig. 7(c) the 2 $\mu$m diameter platelets seem to make two sharp turns as they pass from above the third row of posts to below the row. This behavior was unexpected; to determine its source, we examined the fluid velocity fields calculated by COMSOL for these DLD simulations. Close inspection of the gaps between columns of posts (supplementary material Fig. 8) reveals that COMSOL predicted asymmetric fluid velocity fields in these gaps in the cases where the MOPSA-predicted particle paths were most jagged. For example, the fluid velocity field calculated by COMOSL for the chip in Fig. 7(c) [shown in supplementary

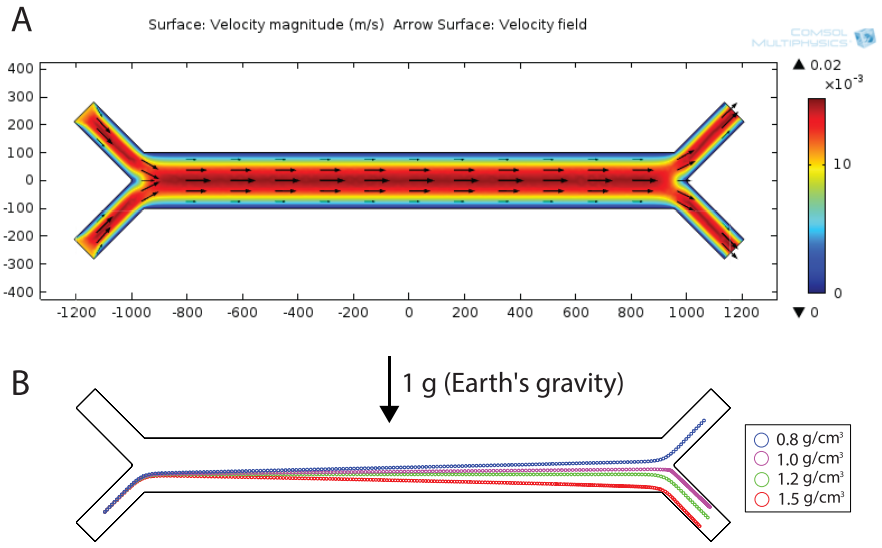


FIG. 8. Using a modified version of MOPSA to simulate the behavior of a microfluidic chip for sorting cells and other particles by their densities. The chip simulated is based on published results[22,23] and consists of two inlets that merge into a single horizontal channel before splitting into two outlets. The chip is oriented on its edge relative to Earth's gravity. (a) Fluid velocity field for the density sorter chip, obtained using COMSOL Multiphysics. (b) After modifying MOPSA using the approach of Haider and Levenspiel,[20] MOPSA predicts the paths followed by four rigid spherical particles of different densities as they enter from the lower input and travel through the chip. As expected, particles with density greater than the fluid density of 1.0 g/cm$^3$ exit the lower outlet, and particles with density less than the fluid density exit the upper outlet. This demonstrates that additional particle properties (like density) can be added to MOPSA when necessary.

material Fig. 8(c)] has a diagonal region of low flow between every two pillars; when MOPSA uses this fluid velocity field to predict the path followed by a particle between the pillars, the asymmetry in the field results in a jagged predicted particle path. These particle trajectories do not seem to affect the overall simulation results (that different-sized particles follow different paths through the DLD chips).

Why was it necessary to increase $\beta$ to successfully simulate DLD chips in MOPSA? One explanation could be our assumption that the presence of the particle will not affect the fluid velocity profile in the chip. In actuality, the presence of a particle *could* increase the hydrodynamic resistance of the DLD chip.[21] This could reduce the fluid velocity in the direction of flow (the $x$ direction) and make the fluid velocity in the direction perpendicular to flow (the $y$ direction) more significant. Increasing $\beta$ from 1 to 1.45 adds additional weight to the $y$ component of the fluid velocity and enables MOPSA to accurately simulate particles in DLD devices, at least for the particle sizes and types considered here. While this value of $\beta$ may not be suitable for all DLD devices or particle sizes and types, it is noteworthy that all six previously published DLD chips we replicated in this work were successfully simulated using the same value for $\beta$ (1.45).

### D. Extending MOPSA to simulate particles with different densities

In its current form, MOPSA is capable of simulating rigid spherical particles in a variety of different microfluidic applications. However, in some cases it may be necessary to add additional physics modules to MOPSA to simulate certain particle properties. For example, microfluidic chips have been demonstrated that sort cells and other particles by their densities.[22,23] Since the current MOPSA assumes that all particles have the same density (and the particles' density equals the density of the fluid around the particles), MOPSA cannot currently be used to simulate the behavior of these density sorter chips.

To demonstrate that additional particle properties can be readily added to MOPSA when needed, we added support for particle density to MOPSA. Our approach uses the work of Haider and Levenspiel[20] to predict the effects of buoyancy on particles. We then used our modified MOPSA to predict the paths followed by rigid spherical particles of four different densities (0.8, 1.0, 1.2, and 1.5 g/cm$^3$) in a microfluidic chip filled with water (1.0 g/cm$^3$) and oriented on its edge relative to Earth's gravity. In Fig. 8, all four particles start at the same location in a lower inlet. As the particles flow along the horizontal channel, our modified MOPSA correctly predicts that the particles with density greater than the fluid density will sink downward and exit via the lower outlet, and the particles with density greater than the fluid density will float upward and exit via the upper outlet. Since different cell types often have different densities,[24] our modified MOPSA may be used to help design chips for sorting cells by their type, an important capability in biological research and medical diagnostics.

## IV. CONCLUSIONS

In this work, we presented MOPSA, an algorithm for optimized particle simulation in microfluidic chips. By treating particles as two-dimensional objects instead of single points and applying corrections when particles interact with channel walls, MOPSA can accurately simulate particle behaviors that particle tracers in existing commercial software tools cannot. Consequently, adopting MOPSA into the design process for microfluidic chips that contain cells, droplets, and other particles should be beneficial.

### A. Limitations of MOPSA and future directions

MOPSA makes several assumptions about the particles it simulates: the particles are circular or spherical, they have the same density as the fluid around them, they have smooth surfaces, they do not interact with other particles, and so on. These assumptions may limit the utility of MOPSA for some applications. For example, in its current form, MOPSA cannot simulate effects of inertia on either the particles or the fluid in a chip; these effects have been used as the basis for particle sorting and focusing in microfluidics.[25] However, for at least some particle sizes and types, MOPSA's assumption of rigid spheres seems to still yield results that are consistent with experimental observations.

For particles that cannot be assumed to be rigid spheres, a number of modeling techniques have been proposed for simulating the behavior of these particles.[26] For example, dissipative particle dynamics[27–29] has been successfully applied to simulate the deformation of red blood cells in DLD chips.[26,30] Zhu *et al.*[31] modeled deformable cells as fluid-filled capsules enclosed by neo-Hookean membranes.[32] Krüger *et al.*[33] used the immersed-boundary method[34] to model membrane dynamics during cell deformation. These and other models can be used in situations where MOPSA fails to accurately predict the behavior of a non-rigid, non-spherical particle; some of them could even be incorporated into future versions of MOPSA.

We also demonstrated that MOPSA can be easily extended to simulate other particle properties like particle density. This same approach can be used to enable MOPSA to support other important properties of particles. For example, by altering the *wallEffect* algorithm to allow cells to stick to channel walls, one could simulate the behavior of microfluidic devices that intentionally capture cells in this manner.[17] Additionally, to include the effects of other forces such as electrostatics and acoustics on a particle's trajectory, additional equations can be added to Line 18 in Algorithm 1. To simulate particle-particle interactions that may be important in applications with high particle concentrations, it may be necessary to periodically recalculate the fluid velocity vector field while including particle positions (and particle-induced drag) in the finite element analysis calculation. In our experience, including particle positions in this manner has a negligible effect on simulation accuracy but significantly increases the required computational time, but it may be necessary for some applications.

Finally, MOPSA is currently a two-dimensional simulation technique and cannot be used to simulate three-dimensional microfluidic channel networks. However, nothing about MOPSA precludes extending the algorithm to three dimensions. This would enable MOPSA to simulate

particle trajectories in emerging 3D-printed microfluidic chips that can have channels in all three spatial dimensions.[35]

## SUPPLEMENTARY MATERIAL

See supplementary material for fluid velocity field for the simulations in Figs. 4, 5, and 7(a)–7(f); results from using COMSOL Multiphysics to simulate the deterministic lateral displacement (DLD) chips in Figs. 7(a)–7(f); closeups of the calculated fluids velocity fields for the DLD chips in Figs. 7(a)–7(f); calculated particle trajectories from MOPSA for the simulations in Figs. 4, 5, and 7(a)–7(f) in MATLAB *MAT* format; and CAD files for the chip designs in Figs. 4, 5, and 7(a)–7(f) in *DXF* format.

## ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation Division of Biological Infrastructure under Award No. DBI-1353974, the National Science Foundation Division of Computer and Communication Foundations program under Award No. CCF-1351115, and the National Science Foundation Division of Industrial Innovation and Partnerships under Award No. IIP-1640757.

[1]S. C. Terry, J. H. Jerman, and J. B. Angell, IEEE Trans. Electron. Devices **26**, 1880 (1979).
[2]G. J. Kost, *Principles & Practice of Point-of-Care Testing* (Lippincott Williams & Wilkins, 2002).
[3]C. W. Shields IV, C. D. Reyes, and G. P. López, Lab Chip **15**, 1230 (2015).
[4]M. Wörner, Microfluid. Nanofluid. **12**, 841 (2012).
[5]M. K. Runyon, C. J. Kastrup, B. L. Johnson-Kerner, T. G. Van Ha, and R. F. Ismagilov, J. Am. Chem. Soc. **130**, 3458 (2008).
[6]J. Siegrist, M. Amasia, N. Singh, D. Banerjee, and M. Madou, Lab Chip **10**, 876 (2010).
[7]L. R. Huang, E. C. Cox, R. H. Austin, and J. C. Sturm, Science **304**, 987 (2004).
[8]K. Loutherback, K. S. Chou, J. Newman, J. Puchalla, R. H. Austin, and J. C. Sturm, Microfluid. Nanofluid. **9**, 1143 (2010).
[9]D. W. Inglis, J. A. Davis, R. H. Austin, and J. C. Sturm, Lab Chip **6**, 655 (2006).
[10]M. Yamada, K. Kano, Y. Tsuda, J. Kobayashi, M. Yamato, M. Seki, and T. Okano, Biomed. Microdev. **9**, 637 (2007).
[11]A. Y. Fu, C. Spence, A. Scherer, F. H. Arnold, and S. R. Quake, Nat. Biotechnol. **17**, 1109 (1999).
[12]X. Wang, S. Chen, M. Kong, Z. Wang, K. D. Costa, R. A. Li, and D. Sun, Lab Chip **11**, 3656 (2011).
[13]H. M. Ji, V. Samper, Y. Chen, C. K. Heng, T. M. Lim, and L. Yobas, Biomed. Microdev. **10**, 251 (2008).
[14]N. Li, D. T. Kamei, and C.-M. Ho, in 2007 2nd IEEE International Conference on Nano/Micro Engineered and Molecular Systems (IEEE, 2007), pp. 932–936.
[15]H. N. Joensson, M. Uhlén, and H. A. Svahn, Lab Chip **11**, 1305 (2011).
[16]See http://www.autodesk.com/techpubs/autocad/acad2000/dxf/ascii_dxf_files_dxf_aa.htm for "ASCII DXF Files" (last accessed September 26, 2016).
[17]S. Nagrath, L. V. Sequist, S. Maheswaran, D. W. Bell, D. Irimia, L. Ulkus, M. R. Smith, E. L. Kwak, S. Digumarthy, A. Muzikansky, P. Ryan, U. J. Balis, R. G. Tompkins, D. A. Haber, and M. Toner, Nature **450**, 1235 (2007).
[18]I. Newton, A. Motte, and N. Chittenden, *Newton's Principia: The Mathematical Principles of Natural Philosophy* (Geo. P. Putnam, 1850).
[19]G. K. Batchelor, *An Introduction to Fluid Dynamics* (Cambridge University Press, 2000).
[20]A. Haider and O. Levenspiel, Powder Technol. **58**, 63 (1989).
[21]M. A. Cartas-Ayala, M. Raafat, and R. Karnik, Small **9**, 375 (2013).
[22]D. Huh, J. H. Bahng, Y. Ling, H.-H. Wei, O. D. Kripfgans, J. B. Fowlkes, J. B. Grotberg, and S. Takayama, Anal. Chem. **79**, 1369 (2007).
[23]J. Song, M. Song, T. Kang, D. Kim, and L. P. Lee, Biomicrofluidics **8**, 064108 (2014).
[24]W. H. Grover, A. K. Bryan, M. Diez-Silva, S. Suresh, J. M. Higgins, and S. R. Manalis, Proc. Natl. Acad. Sci. U.S.A. **108**, 10992 (2011).
[25]D. Di Carlo, Lab Chip **9**, 3038 (2009).
[26]E. Henry, S. H. Holm, Z. Zhang, J. P. Beech, J. O. Tegenfeldt, D. A. Fedosov, and G. Gompper, Sci. Rep. **6**, 34375 (2016).
[27]P. Hoogerbrugge and J. Koelman, Europhys. Lett. **19**, 155 (1992).
[28]P. Espanol and P. Warren, Europhys. Lett. **30**, 191 (1995).
[29]P. Espanol and M. Revenga, Phys. Rev. E **67**, 026705 (2003).
[30]Z. Zhang, E. Henry, G. Gompper, and D. A. Fedosov, J. Chem. Phys. **143**, 243145 (2015).
[31]L. Zhu, C. Rorai, D. Mitra, and L. Brandt, Soft Matter **10**, 7705 (2014).
[32]C. Pozrikidis, *Computational Hydrodynamics of Capsules and Biological Cells*, Chapman and Hall/CRC Mathematical and Computational Biology Series (CRC Press, Boca Raton, 2010).
[33]T. Krüger, D. Holmes, and P. V. Coveney, Biomicrofluidics **8**, 054114 (2014).
[34]C. S. Peskin, Acta Numer. **11**, 479 (2002).
[35]N. Bhattacharjee, A. Urrios, S. Kang, and A. Folch, Lab Chip **16**, 1720 (2016).