Accelerating Simulation of Particle Trajectories in Microfluidic Devices by Constructing a Cloud Database

Junchao Wang†,*, Lingxuan Fu†, Liyang Yu†, Xiwei Huang†, Philip Brisk‡ and William H. Grover§,*
†Ministry of Education Key Laboratory of RF Circuits and Systems. Hangzhou Dianzi University, China.
‡Department of Computer Science and Engineering, University of California Riverside, Riverside, CA, USA
§Department of Bioengineering, University of California Riverside, Riverside, CA, USA,
*Email: junchao@hdu.edu.cn, wgrover@engr.ucr.edu

***Abstract*—Microfluidic cell sorters have shown great potential to revolutionize the current technique of enriching rare cells. In the past decades, different microfluidic cell sorters have been developed by researchers for separating circulating tumor cells, T-cells, and other biological markers from blood samples. However, it typically takes months or even years to design these microfluidic cell sorters by hand. Thus, researchers tend to use computer simulation (usually finite element analysis) to verify their designs before fabrication and experimental testing. Despite this, conducting precision finite element analysis of microfluidic devices is computationally expensive and labor-intensive. To address this issue, we recently presented a microfluidic simulation method that can simulate the behavior of fluids and particles in some typical microfluidic chips instantaneously. Our method decomposes the chip into channels and intersections. The behavior of fluid in each channel is determined by leveraging analogies with electronic circuits, and the behavior of fluid and particles in each intersection is determined by querying a database containing 92,934 pre-simulated channel intersections. While this approach successfully predicts the behavior of complex microfluidic chips in a fraction of the time required by existing techniques, we nonetheless identified three major limitations with this method: (1) the library of pre-simulated channel intersections is unnecessarily large (only 2,072 of 92,934 were used); (2) the library contains only cross-shaped intersections (and no other intersection geometries); and (3) the range of fluid flow rates in the library is limited to 0 to 2 cm/s. To address these deficiencies, in this work we present an improved method for instantaneously simulating the trajectories of particles in microfluidic chips. Firstly, inspired by dynamic programming, our new method optimizes the generation of pre-simulated intersection units and avoids generating unnecessary simulations. Secondly, we constructed a cloud database (*http://cloud.microfluidics.cc*) to share our pre-simulated results and to let users become contributors and upload their simulation results into the cloud database as a benefit to the whole microfluidic simulation community. Lastly, we investigated the impact of different channel angles and different fluid flow rates on predicting the trajectories of particles. We found a wide range of device geometries and flow rates over which our existing simulation results can be extended without having to perform additional simulations. Our method should accelerate the simulation of particles in microfluidic chips and enable researchers to design new microfluidic cell sorter chips more efficiently.**

I. Introduction

Cell sorting has many important applications in both clinics and biological research [2], [3]. Among different cell sorting mechanisms, microfluidic cell sorting devices have shown great potential in enriching rare biological markers

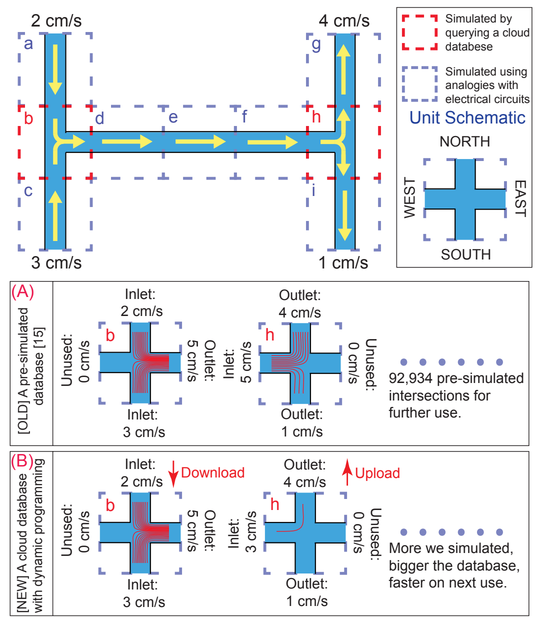

Fig. 1. The comparison of our two methods to accelerate simulation of a microfluidic chip for predicting particle trajectories. (A) Our original method based on memoization [1]. (B) Our new method that was inspired by dynamic programming and a cloud database.

like circulating tumor cells for cancer therapy [4] or T-cells for immunotherapy [5]. In the past decades, researchers have developed different microfluidic cell sorters based on inertial effects [6], [7], [8], pinched flow fractionation [9], [10], deterministic lateral displacement (DLD) [11], [12] and other different principles [13], [14].

During the microfluidic device design process, many researchers may wish to use computer simulations to verify their ideas before fabrication and experimental testing. However, there are two major barriers standing between device ideas and lifelike simulations. First, simulations of microfluidic devices are usually computationally expensive. For instance, a workstation with 32 gigabytes of RAM and a 12-core E5 CPU was necessary to simulate the particle trajectories in a small unit of a DLD device using finite element analysis

2159-3477/18/$31.00 ©2018 IEEE
DOI 10.1109/ISVLSI.2018.00126

(FEA) software (COMSOL Multiphysics) and MOPSA [15] in order to reduce the computational time to an hour. Second, FEA software has a steep learning curve. It can take months of training for a student to learn how to create a successful model of the simulation target and have a basic understanding of the connection between mathematical equations behind the FEA software and real-world physics. These barriers limit the practical usefulness of existing software when simulating microfluidic devices.

To address these issues, our previous work presented a method to efficiently simulate the behavior of fluids and particles in some typical microfluidic chips instantaneously (in approximately one second) [1]. The acceleration is achieved by two steps. Assume we have an "H"-shape microfluidic chip shown in Fig. 1, and we want to simulate the particle trajectories in this chip. First, we decompose the microfluidic chip design into several unit intersections (a - i). Units a, c, d, e, f, g, and i are simple channels; the boundary conditions of these channels can be modeled using the electronic-fluidic analogy and calculated by electronic circuit simulation software such as SPICE [16], [17], [18]. Second, as shown in Fig. 1A, we access a database that contains 5,321,944 pre-simulated trajectories of 92,934 intersections. In low Reynolds number situations, particle trajectories can be expanded from the trajectories of units b and h throughout the whole chip using streamline theory [19]; consequently, we only need to query the database to retrieve pre-simulated results for units b and h. Combining these two steps generates the predicted particle trajectories throughout the device. Constructing the database of pre-simulated intersections required approximately one month of computing time. However, using the database, the simulation of particle trajectories of a microfluidic chip takes approximately one second on a standard laptop, without any noticeable degradation in the accuracy of the simulation.

While our instantaneous simulation method successfully predicts the behavior of complex microfluidic chips in a fraction of the time required by existing techniques, we nonetheless identified three limitations of our technique. These limitations mainly concern the database of pre-simulated intersections. First, the pre-simulated database only contains T- or cross-shaped intersections with 90-degree channel turns; this limits the range of intersection geometries that can be simulated using our technique. Second, the database only contains simulation results with fluid flow rates from 0 to 2 cm/s; it cannot currently be used to predict the behavior of flow rates above 2 cm/s. Third, the time required to construct the pre-simulated database is non-negligible, making it prohibitive to expand its usage to more general chip designs.

In this work, we address these limitations of our previous instantaneous simulation method. We began this work by asking the question: what if we could have a platform to share our pre-simulated database, and every user who simulates a previously unseen intersection contributes the result of the simulation to the database, increasing its utility for others? By leveraging dynamic programming, a method for solving a complex problem by decomposing it into a collection of simpler subproblems [20], we present an online platform which helps researchers take advantage of pre-simulated trajectories in their own microfluidic particle simulation projects. Researchers are able to use our platform through either an application programming interface (API) or a graphical user interface (*http://cloud.microfluidics.cc*). Sharing our simulation database eliminates the need for researchers to construct a brand new database for their own projects and is expected to help researchers obtain reliable particle simulation results instantly for their projects, even if they only need to simulate a small number of microfluidic chips. Researchers are encouraged to upload their simulation results into our cloud database to contribute to this project to help the microfluidic particle simulation community. In theory, as our user base grows, the database of pre-simulated results will grow along with it, and the entire community of users will benefit from faster and more accurate simulations.

To address the other limitations of our previous technique, we also investigate how different channel angles (30° to 180°) at intersections and how different flow rates (Reynolds numbers from 0.1 to 100) affect particle trajectories in the same channel. By sharing this cloud database and generating intersection units on demand, we are confident that simulation of particle trajectories in microfluidic devices will become easier and more efficient in the near future.

II. Cloud Database Design

This work was motivated by the inefficient use of our pre-simulated intersections. In our previous work, we simulated more than 10,000 random microfluidic chips (Fig. 3) using our instantaneous simulation method [1]. In this work, we studied how many of the intersection simulations were actually used while simulating these 10,000 random chips. Table I is the statistical analysis of the usage details of pre-generated intersection units. Although we generated 92,934 intersections with varied boundary conditions, 90,862 of the intersections (97.8%) were never used. Of all generated intersection units, 18 intersection simulations were used more than 1000 times, 205 intersections were used more than 100 times, and 516 intersections were used more than 10 times. Since different intersection units had dramatically different probabilities of being used, it occurred to us that the intersection units should be generated on demand with determinate boundary conditions instead of randomly being generated.

We also introduce a new parameter, called *error tolerance*, to indicate the relative difference between the flow velocity boundary conditions specified by the user and the closest match found in the cloud database. Adjusting the error tolerance allows the user to customize the accuracy required for different applications. Error tolerance is defined in Equation 1.

$$Error\ Tolerance = \left|\frac{V_{W0} - V_{W1}}{V_{W0}}\right| + \left|\frac{V_{N0} - V_{N1}}{V_{N0}}\right| + \left|\frac{V_{S0} - V_{S1}}{V_{S0}}\right| + \left|\frac{V_{E0} - V_{E1}}{V_{E0}}\right| \tag{1}$$

TABLE I

STATISTICAL ANALYSIS OF THE PRE-GENERATED INTERSECTION UNITS IN OUR PREVIOUS WORK [1]

Frequency of intersection unit usage	0	1	$2 \sim 4$	$5 \sim 9$	$10 \sim 99$	$100 \sim 999$	$1000 \sim \infty$
Number of intersection units (total 92,934)	90,862	541	519	273	516	205	18

where V_{W0}, V_{N0}, V_{S0}, V_{E0} indicate the average flow velocity of WEST, NORTH, SOUTH and EAST of the intersection units as requested by the user; and V_{W1}, V_{N1}, V_{S1}, V_{E1} indicate the average flow velocity of WEST, NORTH, SOUTH and EAST of the intersection units in the cloud database.

In dynamic programming, a large problem is decomposed into a series of subproblems. When a subproblem is solved, the solution is stored for future reuse. In our system, the primary problem is to predict particle trajectories in microfluidic devices, and the subproblems are simulations of specific particle trajectories in specific intersection units. The simulated trajectories in each intersection unit are stored in a cloud database for further use. By focusing on subproblems that are actually useful in real-life simulations, as opposed to pre-simulating a large population of randomly-generated intersections, this principle similar to dynamic programming reduces the redundancy in our simulation database and accelerates the simulation process.

Fig. 2 illustrates how we coupled our cloud database with custom simulation projects using the idea of dynamic programming. First, the user determines the flow velocity in the intersections where they intend to simulate the particle trajectories. The details of this step have been described in our previous work [1]. Second, our cloud platform (*http://cloud.microfluidics.cc*) provides an API that helps users find the best match in the cloud database of pre-simulated intersection trajectories. The user sends a POST request to *http://cloud.microfluidics.cc/php/post.php* containing the parameters defined in Listing 1 in JavaScript Object Notation (JSON) format. The cloud platform processes the user request and returns the result to the user, once again in JSON, as defined in Listing 2. In many cases, the results will be suitable for integration into the user's simulation of a complete chip. However, in some cases the cloud platform may not find a suitable match in the database of pre-simulated intersections. In these cases, the user will be encouraged to simulate the trajectories of particles locally and upload the simulation results into our cloud database to better serve the community. To upload a new simulation result, the user sends a JSON-formatted POST request to *http://cloud.microfluidics.cc/php/data_post.php* with the parameters defined in Listing 3. After retrieving or simulating all the particle trajectories in all the intersections, the user can combine and expand the trajectories of individual intersection units into trajectories from the input to the output of the chip [1].

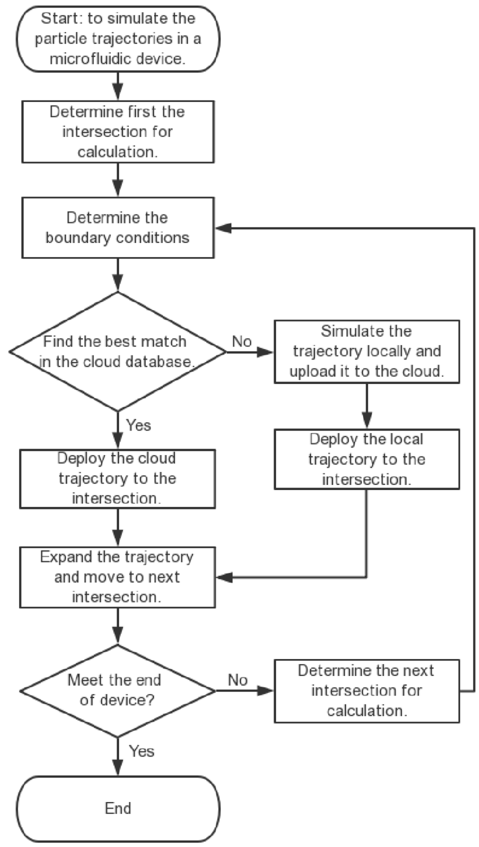

Fig. 2. Flow chart depicting the process of using our cloud database to simulate a user's microfluidic chip design.

Listing 1: Query the database

```
{
    input1 : flow velocity of the WEST boundary of
        the intersection,
    input2 : flow velocity of the NORTH boundary of
        the intersection,
    input3 : flow velocity of the SOUTH boundary of
        the intersection,
    input4 : flow velocity of the EAST boundary of
        the intersection,
    accuracy : Error tolerance,
    dp : particle diameter,
}
```

Listing 2: Pre-simulated particle trajectories

```
{
    "res_code" : 0 for failure & 1 for success,
    "msg" : message,
    "res" : [{
        "id" : ID of the trajectory,
        "start_position_x" : initial position of the
             particle in x direction,
```

```
        "start_position_y" : initial position of the
             particle in y direction,
        "end_position_x" :  final position of the
            particle in x direction,
        "end_position_y" :  final position of the
            particle in y direction,
        }]
}
```

Listing 3: Uploading the simulated trajectories

```
{
    input1 : flow velocity of the WEST boundary of
        the intersection,
    input2 : flow velocity of the NORTH boundary of
        the intersection,
    input3 : flow velocity of the SOUTH boundary of
        the intersection,
    input4 : flow velocity of the EAST boundary of
        the intersection,
    dp : particle diameter,
    length : The number of trajectories uploaded,
    data0 : initial position of the particle in x
        direction,
    data1 : initial position of the particle in y
        direction,
    data2 : final position of the particle in x
        direction,
    data3 : final position of the particle in y
        direction,
    ...

    file : binary files describing the details of
        trajectories if possible,
}
```

III. Validating Fast Simulating of Particle Trajectories in Random Microfluidic Chips

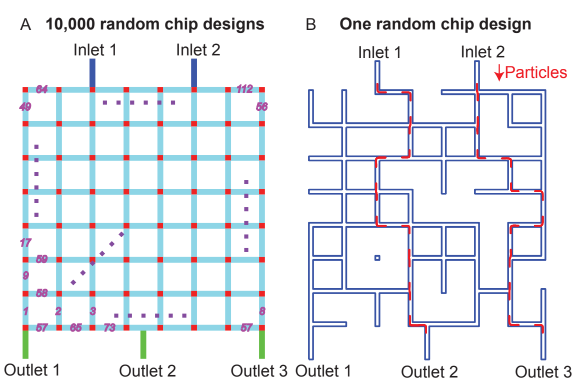

Fig. 3. (A) The schematic of random microfluidic chips [21]. (B) Results from using the principle of dynamic programming and our cloud database to predict the paths of two 1 μm diameter particles traveling through a random microfluidic chip.

To demonstrate our method and cloud platform on a more complex chip design, we used the new method to predict the trajectories of 1 μm diameter particles in 10,000 randomly-generated microfluidic chips [21]. As shown in Fig. 3, these random microfluidic chips each have two inlets and three outlets. In our simulations, particles start in the center of each inlet channel and end at one outlet.

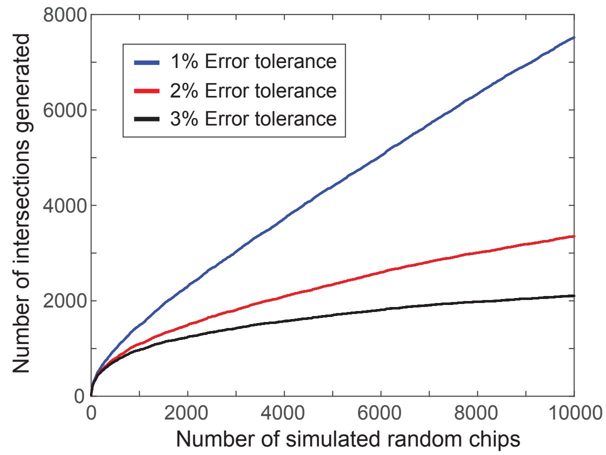

Fig. 4. Statistical analysis of the generated intersection units using the principle of dynamic programming and cloud database method when simulating 10,000 random microfluidic chips with 1%, 2% and 3% error tolerance settings.

The cloud database was reset prior to each simulation. Each simulation was performed three times with different error tolerance settings (1%, 2% and 3%); the simulation results are summarized in Fig. 4. The three error tolerance settings generated 7,519, 3,351 and 2,102 new intersection units respectively. In each case, as the number of simulated random microfluidic chips increased, new intersection units were generated rapidly at first. After generating a certain number of intersection units, the generation rate decreased, indicating greater reuse of existing pre-simulated intersection units over time. Increasing the error tolerance decreases the number of new intersection units generated during the simulation process, which is consistent with our prediction. Compared with the intersection units generated in our previous work (Table I), every intersection unit generated by our new method was used at least once. Furthermore, even for the most-stringent 1% error tolerance case, only 7,519 intersection units were generated instead of 92,934. The computational time required to generate the database was therefore reduced by 91.9%, 96.4% and 97.7% with 1%, 2% and 3% error tolerance, respectively, when compared to our previous work. Additionally, users may benefit from the ability to specify their own expectations about error tolerance, which is handled seamlessly by the querying mechanism that we have integrated into our cloud database.

IV. Removing Limitations on the Device Geometries That Can Be Simulated

As noted earlier, our pre-simulated intersection database only contains simulations for channel intersections with (1) 90° turns and (2) fluid flow rates between 0 and 2 cm/s, so the database could only be used for microfluidic chips that adhere to these constraints. In this work, we have successfully removed these limitations. We accomplished this by building FEA models of microfluidic channels that have one inlet, two outlets, and internal angles of 30°, 60°, 90°, 120°, 150° and 180°. We then simulated flow through each intersection using inlet flow rates that resulted in four different values for the Reynolds number: 0.1, 1, 10, and 100. We solved the fluid velocity field of the chip using the Laminar Flow physics module in COMSOL Multiphysics and a stationary solver; the simulation used the "Extremely Fine" mesh setting. We used the Particle Tracing for Fluid Flow physics module to predict particle trajectories across the entire chip. We added a "Drag Force" boundary condition to the entire chip, and a particle "Inlet" boundary condition with initial position "Uniform Distribution" and we added a 1.0 μm particle diameter to all inlets in the Laminar Flow module. We assigned "Outlet" boundary conditions to the outlets in the Laminar Flow module; the remaining boundaries were walls ("freeze" boundary condition). We set the number of particles per release to 20.

Fig. 5 shows the simulation results. The relative end position of 20 particles are near-identical at Reynolds numbers of 0.1, 1 and 10, across all six different intersection angles (except the case of Re = 10 and angle = 30°). The average relative differences are 1.3%, 2.2% and 5.2% when Re is equal to 0.1, 1 and 10, respectively. At a Reynolds number of 100, the angle of intersection begins to impact the particle trajectories, and the particles exit the intersections at substantially different locations in the six different angles. When channel angles equal 30° and 180°, the increase of Re has greater impact on the trajectories of particles. In the 30° and 180° cases, the simulations predict that the middle particles tend to become stuck in the intersections when Re equals 10 and 100.

These results show that even though our cloud database has limited pre-simulated intersection units, users are nonetheless able to convert the pre-simulated results into new simulation results (for Reynolds numbers less than 10, and channel angles between 60° and 180°) and obtain reliable simulation results without actually performing new finite element analyses. This significantly increases the range of microfluidic device geometries and flow rates that can be instantaneously simulated using our method. However, there are still some intersection geometries and flow rates that cannot be simulated using our improved method. In these cases, our cloud database can ultimately allow users to upload their own simulation results containing currently-unsupported angles and flow rates, thereby further extending the range of devices that can be simulated instantaneously.

V. Conclusions

In this work, we constructed a cloud database and applied the principle of dynamic programming to accelerate the simulation of particle trajectories in microfluidic devices. Compared to our previous method, the efficiency of our new method arises from three key innovations: 1. Only simulate the intersection units which will be used at least once; 2. Utilize a cloud database that will help the microfluidics community accelerate predicting the path of particles in their own projects; 3. Enable users of our platform to submit their own simulation results to expand our pre-simulated database. The original limitations of our method (only supporting cross-shape intersections and a limited range of fluid flow rates) were investigated as well. Our simulation results show that these limitations can be negligible at low Reynolds number, which is one of the natural properties of microfluidics. This method reduces the barriers to simulating particle trajectories in microfluidic chips and should ultimately enable researchers to design new microfluidic cell sorting devices more efficiently.

Acknowledgment

This work was supported in part by Hangzhou Dianzi University seed fund for JW, and National Science Foundation awards #1351115, #1353974, #1536026, #1640757, and #1740052.

References

[1] J. Wang, V. G. J. Rodgers, P. Brisk, and W. H. Grover, "Instantaneous simulation of fluids and particles in complex microfluidic devices," *PLOS ONE*, vol. 12, no. 12, pp. 1–14, 12 2017. [Online]. Available: https://doi.org/10.1371/journal.pone.0189429

[2] N. M. Karabacak, P. S. Spuhler, F. Fachin, E. J. Lim, V. Pai, E. Ozkumur, J. M. Martel, N. Kojic, K. Smith, P.-i. Chen *et al.*, "Microfluidic, marker-free isolation of circulating tumor cells from blood samples," *Nature protocols*, vol. 9, no. 3, p. 694, 2014.

[3] L. A. Herzenberg, D. W. Bianchi, J. Schröder, H. M. Cann, and G. M. Iverson, "Fetal cells in the blood of pregnant women: detection and enrichment by fluorescence-activated cell sorting," *Proceedings of the National Academy of Sciences*, vol. 76, no. 3, pp. 1453–1455, 1979.

[4] V. Plaks, C. D. Koopman, and Z. Werb, "Circulating tumor cells," *Science*, vol. 341, no. 6151, pp. 1186–1188, 2013.

[5] C. A. Klebanoff, S. A. Rosenberg, and N. P. Restifo, "Prospects for gene-engineered t cell immunotherapy for solid cancers," *Nature medicine*, vol. 22, no. 1, p. 26, 2016.

[6] S. S. Kuntaegowdanahalli, A. A. S. Bhagat, G. Kumar, and I. Papautsky, "Inertial microfluidics for continuous particle separation in spiral microchannels," *Lab on a Chip*, vol. 9, no. 20, pp. 2973–2980, 2009.

[7] L. Wu, G. Guan, H. W. Hou, A. A. S. Bhagat, and J. Han, "Separation of leukocytes from blood using spiral channel with trapezoid cross-section," *Analytical chemistry*, vol. 84, no. 21, pp. 9324–9331, 2012.

[8] H. W. Hou, M. E. Warkiani, B. L. Khoo, Z. R. Li, R. A. Soo, D. S.-W. Tan, W.-T. Lim, J. Han, A. A. S. Bhagat, and C. T. Lim, "Isolation and retrieval of circulating tumor cells using centrifugal forces," *Scientific reports*, vol. 3, p. 1259, 2013.

[9] M. Yamada, M. Nakashima, and M. Seki, "Pinched flow fractionation: continuous size separation of particles utilizing a laminar flow profile in a pinched microchannel," *Analytical chemistry*, vol. 76, no. 18, pp. 5465–5471, 2004.

[10] A. A. S. Bhagat, H. W. Hou, L. D. Li, C. T. Lim, and J. Han, "Pinched flow coupled shear-modulated inertial microfluidics for high-throughput rare blood cell separation," *Lab on a Chip*, vol. 11, no. 11, pp. 1870–1878, 2011.

[11] L. R. Huang, E. C. Cox, R. H. Austin, and J. C. Sturm, "Continuous particle separation through deterministic lateral displacement," *Science*, vol. 304, no. 5673, pp. 987–990, 2004.

[12] S. H. Holm, J. P. Beech, M. P. Barrett, and J. O. Tegenfeldt, "Separation of parasites from human blood using deterministic lateral displacement," *Lab on a Chip*, vol. 11, no. 7, pp. 1326–1332, 2011.

[13] C. W. Shields IV, C. D. Reyes, and G. P. López, "Microfluidic cell sorting: a review of the advances in the separation of cells from debulking to rare cell isolation," *Lab on a Chip*, vol. 15, no. 5, pp. 1230–1249, 2015.

[14] J. Autebert, B. Coudert, F.-C. Bidard, J.-Y. Pierga, S. Descroix, L. Malaquin, and J.-L. Viovy, "Microfluidic: an innovative tool for efficient cell sorting," *Methods*, vol. 57, no. 3, pp. 297–307, 2012.

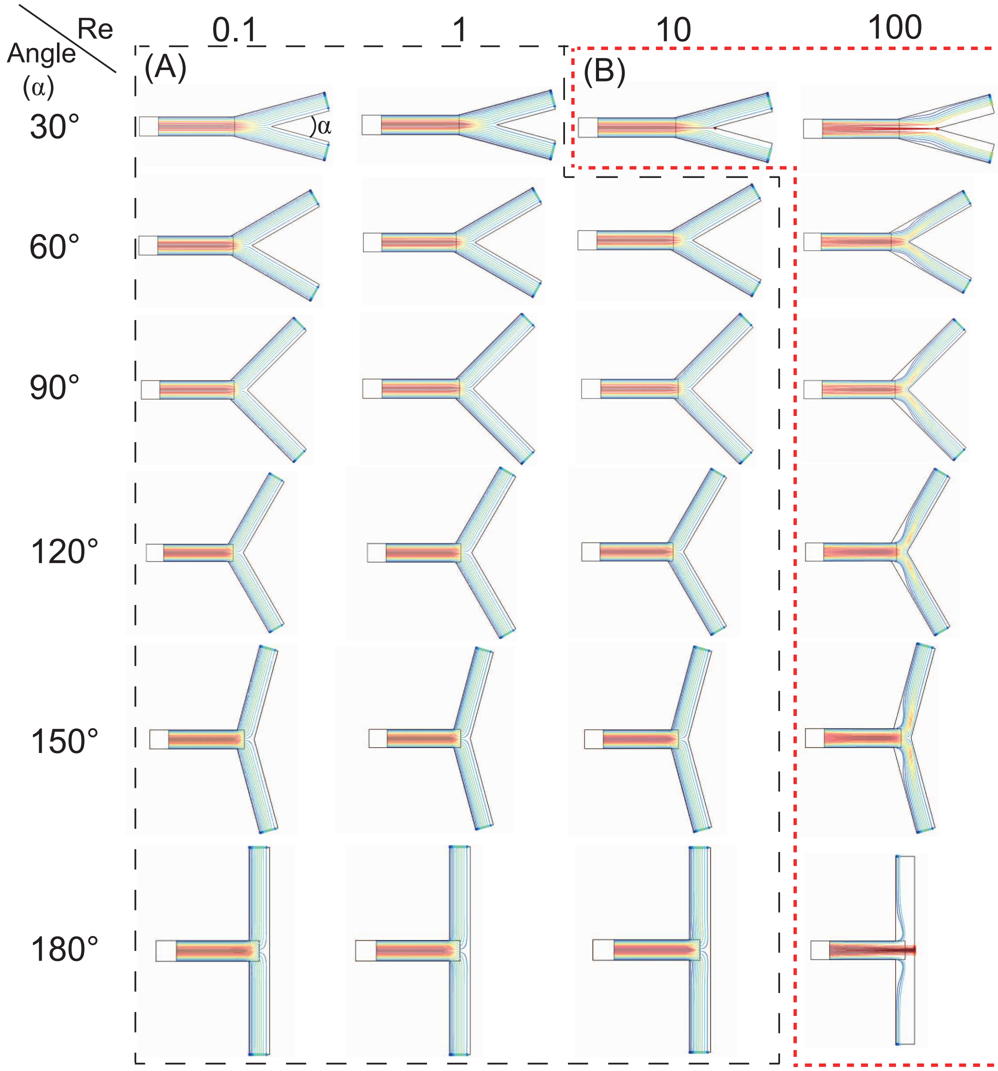

Fig. 5. Simulation results of microfluidic intersections with different Reynolds numbers (0.1, 1, 10 and 100) and different intersection angles (30°, 60°, 90°, 120°, 150° and 180°). (A) At low Reynolds numbers (less than 10) and for channel angles between 60° and 180°, the predicted end positions of the particles are basically identical. This suggests that unit intersection simulations in our cloud database can be used to predict the behavior of microfluidic chips with a wide range of flow rates and intersection geometries. (B) At a Reynolds number of 100, the angle of intersection begins to impact the particle trajectories, and the particles exit the intersections at substantially different locations in the six different angles. When Re = 10 and angle = 30°, the result is similar to Re = 100 and angle = 30° case.

[15] J. Wang, V. G. Rodgers, P. Brisk, and W. H. Grover, "MOPSA: A microfluidics-optimized particle simulation algorithm," *Biomicrofluidics*, vol. 11, no. 3, p. 034121, 2017.
[16] L. W. Nagel, "Spice-simulation program with integrated circuit emphasis," *Memo No.. ERL-M382, Electronics Research Laboratory, Univ. of California, Berkeley*, 1973.
[17] D. Kim, N. C. Chesler, and D. J. Beebe, "A method for dynamic system characterization using hydraulic series resistance," *Lab on a Chip*, vol. 6, no. 5, pp. 639–644, 2006.
[18] D. J. Beebe, G. A. Mensing, and G. M. Walker, "Physics and applications of microfluidics in biology," *Annual review of biomedical engineering*, vol. 4, no. 1, pp. 261–286, 2002.
[19] L. Han, "Hydrodynamic entrance lengths for incompressible laminar flow in rectangular ducts," *Journal of Applied Mechanics*, vol. 27, no. 3, pp. 403–409, 1960.
[20] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 1, no. 2.
[21] J. Wang, P. Brisk, and W. H. Grover, "Random design of microfluidics," *Lab on a Chip*, vol. 16, no. 21, pp. 4212–4219, 2016.