Software Impacts 14 (2022) 100440

Contents lists available at ScienceDirect

Software Impacts

journal homepage: www.journals.elsevier.com/software-impacts

Original software publication

MMFT Droplet Simulator: Efficient Simulation of Droplet-based Microfluidic Devices

Gerold Fink [a,*], Florina Costamoling [a], Robert Wille [b,c]

[a] *Johannes Kepler University Linz - Institute for Integrated Circuits, 4040 Linz, Austria*
[b] *Technical University of Munich - Chair for Design Automation, 80333 Munich, Germany*
[c] *Software Competence Center Hagenberg GmbH (SCCH), 4232 Hagenberg, Austria*

ARTICLE INFO

Keywords:
Microfluidic
Droplet
Simulation

ABSTRACT

Microfluidic devices have found many great applications in medicine, (bio-)chemistry, pharmacology, etc. Unfortunately, their design process is still in its infancy, and frequently results in a costly and time-consuming "trial-and-error" approach, where designs are derived by hand. In order to prevent this, design automation methods and simulation tools can be utilized that aid designers during the whole design phase. In this article, we present such a simulation tool that allows to simulate the behavior of droplet-based microfluidic devices and, by this, allows to validate the functionality of the device, even before the first prototype is fabricated.

Code metadata

Current code version	v1.0.0
Permanent link to code/repository used for this code version	https://github.com/SoftwareImpacts/SIMPAC-2022-234
Permanent link to Reproducible Capsule	https://codeocean.com/capsule/9337788/tree/v1
Legal Code License	MIT License
Code versioning system used	git
Software code languages, tools, and services used	C++
Compilation requirements, operating environments & dependencies	cmake ≥ 3.21, C++-17 compatible compiler
If available Link to developer documentation/manual	
Support email for questions	microfluidics.cda@xcit.tum.de

1. Introduction

Microfluidics is an emerging field which aims to shrink bulky and expensive lab equipment onto a single microfluidic chip. These chips are therefore often called *Lab-on-a-Chip* (LoC) and found many great applications in domains such as medicine, (bio-)chemistry, biology, pharmacology, etc. [1–3]. This miniaturization and integration into a single chip allows to automate and parallelize lab operations/experiments and comes with many benefits, such as reduced volumes of samples/reagents (which are often times costly or rare), higher throughput, shorter reaction times, reduced costs, etc.

Especially droplet-based microfluidic devices (which are a special kind of LoCs) are a promising technology. Here, droplets which contain certain samples/reagents are transported through closed microchannels by a second immiscible fluid, the so-called continuous phase. By routing these droplets through different modules on the LoC, different operations can be conducted, such as heating, mixing, incubating etc., which allows to process a broad range of (bio-)chemical experiments [4–6].

Despite these promises, the design process of microfluidic devices is still in its infancy and mostly conducted manually, which is especially critical, since designing such LoCs is a rather complex task where a huge number of physical parameters need to be considered. Unfortunately, this often results in a "trial-and-error" approach, where a design gets fabricated and then tested whether it satisfies the desired functionality. If it does not show the intended behavior, the design is revised and the fabrication/testing is conducted again – resulting in a time-consuming and costly debugging loop. To overcome this, simulation methods and design automation tools such as [7–9] can be utilized to aid designers during the design phase, before even the first

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

[*] Corresponding author.
E-mail addresses: gerold.fink@jku.at (G. Fink), florina.costamoling@jku.at (F. Costamoling), robert.wille@tum.de (R. Wille).

https://doi.org/10.1016/j.simpa.2022.100440
Received 29 October 2022; Accepted 1 November 2022

prototype is manufactured. This allows to break these costly debugging loops and reduce time-to-market.

In this article, we present such a simulation tool for droplet-based microfluidic devices that allows to simulate the paths of droplets inside a LoC and, by this, capture the behavior of the device. Hence, this droplet simulator is able to verify whether a design works as intended, without the need of an actual fabricated device. Moreover, the setup time as well as the total computation time for a simulation is very short, so different designs can be tested in negligible run-time. This becomes handy when it comes to quick design explorations in earlier design stages. In the next section we discuss the functionality of the simulator in more detail.

2. Description and features

In order to describe the behavior of a microfluidic device, a proper physical model is required. Here often *Computational Fluid Dynamics* (CFD) simulations are used, which are very accurate but, unfortunately, also computational expensive [10]. Hence, CFD simulations are only suited for evaluating smaller designs, or designs that are already in later stages of the design phase, where the additional overhead of CFD simulations is acceptable.

In contrast, we are using the one-dimensional (1D) analysis model (or short 1D-model) [11], which is a more abstract model and, thus, needs less resources in terms of computational power and setup time, but is also not as precise as CFD simulations. Hence, it is perfectly suited for early design explorations and rough verification tests of existing designs, where efficiency is more important than accuracy.

2.1. 1D analysis model

The 1D-model itself can be applied in scenarios with a fully developed, laminar, viscous and incompressible flow, which is often the case in microfluidic devices. In such conditions, the relationship between the pressure drop ΔP, the volumetric flow rate Q and the hydrodynamic resistance R inside a microfluidic channel can be described by *Hagen–Poiseuille's law* [11,12], as $\Delta P = Q\ R$. For rectangular channels with a certain width w, height h and length l, the value of the hydrodynamic resistance can be computed by (assuming $h/w < 1$)

$$R(l) = 12 \left[1 - \frac{192\,h}{\pi^5\,w} \tanh\left(\frac{\pi\,w}{2\,h}\right)\right]^{-1} \frac{\mu_c\,l}{w\,h^3}\ , \tag{1}$$

where μ_c is the viscosity of the continuous phase. Additionally, a droplet with the length l_d inside a channel increases its resistance by $R_\mathrm{d} = b\ R(l_\mathrm{d})$, where b is a factor between 2 and 5 according to [13].

Furthermore, Hagen–Poiseuille's law is the equivalent to Ohm's law in the electrical domain, which allows to convert hydrodynamic components into electrical ones, i.e., channels are converted to electrical resistances and syringe/pressure pumps are converted to current/voltage sources. This eventually allows to construct an equivalent electrical network, which has the benefit that well-known algorithms and methods from the electrical domain can be applied in order to compute the flow rates and pressure drops of each channel inside the microfluidic device. Hence, this equivalent electrical network is crucial for simulating the behavior of such devices and acts as the basis for the droplet simulator reviewed next.

2.2. Simulator workflow

With the equivalent electrical network on hand, the whole flow state of the device (i.e., the pressure drops and flow rates of the channels) can be computed at any given time, which is necessary to predict the behavior of droplets inside the device. Basically, the proposed simulator does this in an event-based approach, i.e., the flow state is assumed to be constant until and event gets triggered. Hence, the movement of the droplets can then be predicted between these events. More precisely, the workflow of the simulator can be described in the following six steps:

1. *Initialization:* The first step is to initialize the simulator by capturing all user inputs (i.e., structure of the microfluidic device, dimensions of channels, fluid properties, droplet injections, etc.) in order to generate a proper starting point for the simulation. After that, the following steps are performed in a loop.
2. *Current flow state:* With the help of the equivalent electrical network, the simulator computes the current flow state (i.e., the pressure drops and flow rates of each channel) of the device. This flow state is important in the next steps, since it is required to compute events as well as to determine the movement of droplets.
3. *Next event:* In this step, the simulator computes all events that could possible be triggered next. An event itself is basically an incident that changes the current flow state and can be triggered, e.g., by injecting a droplet, when a droplet switches channels, or when a droplet leaves the network. With all these events on hand, the simulator now selects the event which will happen next in time as the next event.
4. *Move droplets:* Since the time the next even will happen is known, the simulator is able to move all droplets accordingly in time by utilizing the (in this moment) constant flow state from Step 2.
5. *Perform Event:* After moving the droplets, the simulator now performs the corresponding event, e.g., injecting a new droplet at a given location. However, this also means that the current flow state is now invalid, since (in general) an event changes the underlying equivalent electrical network. Hence, the flow state must be updated again during the next iteration at Step 2.
6. *Termination condition:* If a termination criterion is reached (e.g., all droplets left the network), the simulator stops, otherwise it continues with Step 2 again.

Overall this allows to efficiently capture the behavior of a droplet-based microfluidic device and allows to easily obtain the paths of the droplets, since each system state (droplet positions, flow state, etc.) is stored at every time step.

2.3. Example

To showcase the usage of the simulator, a small device as illustrated in Fig. 1 is considered. It consists of 6 channels (indicated by c_i), 5 nodes that connect the channels (indicated by n_i), as well as a syringe pump and a sink (where the droplets leave the device). Additionally, a droplet (marked in red) should be injected into the channel c_1.

In order to create and simulate the device the following steps have to be conducted, which are also listed in the corresponding code sample shown in Fig. 2:

1. Instantiate the simulator.
2. Add one or more pumps. Here, the start and end node as well as the pressure/flow rate have to be specified. Please note, that the nodes are implicitly defined and do not have to be created by the user.
3. Add channels with their start and end node as well as their dimensions (height, width, length).
4. Define nodes as sinks (where droplets leave the device) and define a ground node, which acts as a reference node for the pressure values in all nodes.
5. Add fluids with certain viscosities and densities.
6. Define the fluid which acts as the continuous phase.
7. Add droplets that should be injected into the device. Here the fluid and the volume of the droplet must be specified. Additionally, the time and position (i.e., in which channel at which relative position inside this channel) the droplet should be injected also have to be declared.
8. Run the simulation and print the results.

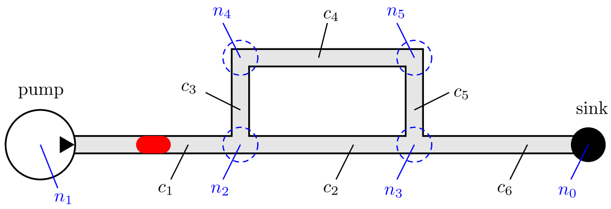

Fig. 1. Microfluidic device example.

Overall, the effort to setup the initial simulation is very low and changing certain values to simulate a different design/version is a matter of seconds – a huge advantage during the design phase of such devices.

3. Impact overview

The droplet simulator first started out with a Java implementation [14] and was then re-implemented in C++ into its current state. During the years, it has served as backbone for many works, where the most prominent ones are listed in the following.

- A demonstration for the benefits of the droplet simulator is the design improvement of the device proposed in [6], which is able to screen drug compounds that inhibit the tau-peptide aggregation. Here, the original design was created by a "trial-and-error" approach, which took about 30 days, required six different prototypes, and resulted in costs of about $1200. By utilizing the proposed simulator, it was shown that the final design could be derived within a single day and, thus, only a single prototype would be needed, which eventually would reduce the costs to $200 [15].
- The simulator was crucial during several design tasks regarding different architectures for microfluidic networks[1], such as a ring network [16], two-dimensional network [17], as well as a pressure-controlled network [18].
- The simulator was also used for evaluating [19] and improving [20] the robustness of microfluidic devices, i.e., how robust a design is against defects that occur due to imperfections in the manufacturing processes or due to swelling of the used materials. Here, especially the fast computational time was of great advantage.

4. Conclusion

In this article, we presented an easy-to-use droplet simulator, which allows to efficiently simulate and validate the behavior of droplet-based microfluidic devices and, by this, aid designers during the whole design phase of such devices. As a result, the functionality of corresponding designs can already be verified, even before a first prototype is fabricated – reducing costs and saving time. The simulator is available at https://github.com/cda-tum/mmft-droplet-simulator.git and is part of the *Munich MicroFluidics Toolkit* (MMFT) developed at the Technical University of Munich.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

```
#include <iostream>
#include "droplet-simulator/Results.h"
#include "droplet-simulator/Simulator.h"

int main(int argc, char const* argv[]) {
    auto n0 = 0, n1 = 1, n2 = 2, n3 = 3, n4 = 4, n5 = 5;
    // 1. Instantiate the simulator
    droplet::Simulator sim;
    // 2. Add pumps
    auto flowRate = 3e-11;
    auto pump = sim.addFlowRatePump(n0, n1, flowRate);
    // 3. Add channels
    auto cWidth = 100e-6, cHeight = 30e-6, cLength = 1000e-6;
    auto c1 = sim.addChannel(n1, n2, cHeight, cWidth, cLength);
    auto c2 = sim.addChannel(n2, n3, cHeight, cWidth, cLength);
    auto c3 = sim.addChannel(n2, n4, cHeight, cWidth, 0.5 * cLength);
    auto c4 = sim.addChannel(n4, n5, cHeight, cWidth, cLength);
    auto c5 = sim.addChannel(n5, n3, cHeight, cWidth, 0.5 * cLength);
    auto c6 = sim.addChannel(n3, n0, cHeight, cWidth, cLength);
    // 4. Define sinks & ground node
    sim.addSink(n0);
    sim.addGround(n0);
    // 5. Add fluids
    auto fluid0 = sim.addFluid(1e-3, 1e3);
    auto fluid1 = sim.addFluid(3e-3, 1e3);
    // 6. Define the continuous phase
    sim.setContinuousPhase(fluid0);
    // 7. Add droplets
    auto dropletVolume = 1.5 * cWidth * cWidth * cHeight;
    auto droplet0 = sim.addDroplet(fluid1, dropletVolume, 0.0, c1, 0.5);
    // 8. Run the simulation & print results
    auto result = sim.simulate();
    std::cout << result.toJson(4) << std::endl;
}
```

Fig. 2. Code example.

Acknowledgments

This work has partially been supported by the FFG, Austria project AUTOMATE (project number: 890068) as well as by BMK, BMDW, and the State of Upper Austria in the frame of the COMET Programme managed by FFG.

References

[1] G.M. Whitesides, The origins and the future of microfluidics, Nature 442 (7101) (2006) 368–373.

[2] D. Mark, S. Haeberle, G. Roth, F. von Stetten, R. Zengerle, Microfluidic Lab-on-a-Chip platforms: requirements, characteristics and applications, Chem. Soc. Rev. 39 (3) (2010) 1153–1182.

[3] P.S. Dittrich, A. Manz, Lab-on-a-chip: microfluidics in drug discovery, Nat. Rev. Drug Discov. 5 (3) (2006) 210.

[4] H. Gu, M.H. Duits, F. Mugele, Droplets formation and merging in two-phase flow microfluidics, Int. J. Mol. Sci. 12 (4) (2011) 2572–2597.

[5] S.-Y. Teh, R. Lin, L.-H. Hung, A.P. Lee, Droplet microfluidics, Lab A Chip 8 (2008) 198–220.

[6] X. Chen, C.L. Ren, A microfluidic chip integrated with droplet generation, pairing, trapping, merging, mixing and releasing, RSC Adv. 7 (27) (2017) 16738–16750.

[7] A. Grimmer, P. Frank, P. Ebner, S. Häfner, A. Richter, R. Wille, Meander designer: Automatically generating meander channel designs, Micromach. – J. Micro/Nano Sci. Devices Appl. 9 (12) (2018).

[8] G. Fink, T. Mitteramskogler, M.A. Hintermüller, B. Jakoby, R. Wille, Automatic design of microfluidic gradient generators, IEEE Access 10 (2022) 28155–28164.

[9] G. Fink, P. Ebner, R. Wille, Comprehensive and accessible channel routing for microfluidic devices, in: Design, Automation and Test in Europe, DATE, 2022.

[10] T. Glatzel, C. Litterst, C. Cupelli, T. Lindemann, C. Moosmann, R. Niekrawietz, W. Streule, R. Zengerle, P. Koltay, Computational fluid dynamics (CFD) software tools for microfluidic applications–a case study, Comput. & Fluids 37 (3) (2008) 218–235.

[11] K.W. Oh, K. Lee, B. Ahn, E.P. Furlani, Design of pressure-driven microfluidic networks using electric circuit analogy, Lab A Chip 12 (3) (2012) 515–545.

[12] H. Bruus, Theoretical Microfluidics, Vol. 18, Oxford university press Oxford, 2008.

[13] T. Glawdel, C.L. Ren, Global network design for robust operation of microfluidic droplet generators with pressure-driven flow, Microfluid. Nanofluid. 13 (3) (2012) 469–480.

[14] A. Grimmer, M. Hamidović, W. Haselmayr, R. Wille, Advanced simulation of droplet microfluidics, J. Emerg. Technol. Comput. Syst. (2019).

[1] Microfluidic networks in this context refers to droplet-based devices, where the paths of droplets are not predefined, but can be controlled by different mechanisms.

[15] A. Grimmer, X. Chen, M. Hamidović, W. Haselmayr, C.L. Ren, R. Wille, Simulation before fabrication: a case study on the utilization of simulators for the design of droplet microfluidic networks, RSC Adv. 8 (2018) 34733–34742, URL http://dx.doi.org/10.1039/C8RA05531A.

[16] G. Fink, M. Hamidović, R. Wille, W. Haselmayr, Passive droplet control in two-dimensional microfluidic networks, Trans. Mol. Biol. Multi-Scale Commun. (2020).

[17] G. Fink, M. Hamidović, W. Haselmayr, R. Wille, Automatic design of droplet-based microfluidic ring networks, Trans. Comput.-Aided Des. Integr. Circuits Syst. (2020).

[18] G. Fink, M. Hamidović, W. Haselmayr, R. Wille, A concept towards pressure-controlled microfluidic networks, in: Int'L Symp. on Design and Diagnostics of Electronic Circuits and Systems, IEEE, 2022, pp. 118–123.

[19] G. Fink, A. Grimmer, M. Hamidović, W. Haselmayr, R. Wille, Robustness analysis for droplet-based microfluidic networks, Trans. Comput.-Aided Des. Integr. Circuits Syst. (2019).

[20] G. Fink, P. Ebner, S. Poddar, R. Wille, Improving the robustness of microfluidic networks, in: Asia and South Pacific Design Automation Conference, IEEE, 2022, pp. 68–73.