Error Recovery in Digital Microfluidics for Personalized Medicine*

Mohamed Ibrahim and Krishnendu Chakrabarty

Department of Electrical and Computer Engineering
Duke University, Durham, NC 27708, USA
{mohamed.s.ibrahim, krishnendu.chakrabarty}@duke.edu

***Abstract*—Due to its emergence as an efficient platform for point-of-care clinical diagnostics, design optimization of digital-microfluidic biochips (DMFBs) has received considerable attention in recent years. In particular, error recoverability is of key interest in medical applications due to the need for system reliability. Errors are likely during droplet manipulation due to defects, chip degradation, and the lack of precision inherent in biochemical experiments. We present an illustrative survey on recently proposed techniques for error recovery. The parameters of the error-recovery design space are shown and evaluated for these schemes. Next, we make use of these evaluations to describe how they can guide error recovery in DMFBs. Finally, an experimental case study is presented to demonstrate how an error-recovery scheme can be applied to real-life biochips.**

I. Introduction

Point-of-care (POC) tests have the potential to improve the management and treatment of infectious diseases, especially in resource-limited settings in developing countries where health care infrastructure is inadequate [1]. In these settings, POC tests can be simply used at the primary-care level with no laboratory infrastructure [2]. In addition, POC tests can potentially empower patients to self-test in the privacy of their homes, especially for diseases such as HIV that carry some degree of social stigma [3]. These issues have motivated researchers from various domains to focus on developing robust technologies for POC diagnostics. Nowadays, clinical pathologists work in cooperation with biochemists to develop new sequences and protocols for obtaining rapid screening results [4]. New substrate materials and biosensors facilitate the miniaturization of sample preparation and automation of POC sequencing on a chip [5]. An example of an emerging technology that has achieved remarkable success in miniaturizing POC testing is digital microfluidics, resulting in digital-microfluidic biochips (DMFBs) [6]. With this application-technology coupling, the global health community can take advantage of diagnostics-on-a-chip (DOC).

Digital-microfluidic biochip technology, which allows us to manipulate droplets of picoliter volumes under program control on a patterned electrode array, is revolutionizing laboratory procedures not only for point-of-care clinical diagnostics [6], but also for many other applications such as environmental monitoring [7] and drug discovery [8]. Over the past decade, there has been a considerable amount of research on various aspects of automated biochip design and optimization [9], [10], including techniques for architectural-level synthesis [11], [12], module placement [12], and droplet routing [13]–[16]. The majority of these techniques address offline synthesis for application-specific DMFBs. A major stumbling block in the monitoring and controlling of diseases/contaminations via POC systems is the lack of adaptive and reliable diagnostic tests that can recover from unexpected errors. Moreover, due to the inherent randomness and complexity of the component interactions that are ubiquitous in biochemistry, it is necessary to verify the correctness of on-chip fluidic interactions during bioassay execution [17]. As a result, in order to enhance the market share of POC clinical diagnostic instrumentation, an efficient design of a DOC requires careful consideration of error recoverability.

A DOC device is said to have a failure if its operation does not match its specified behavior. In order to detect defects using electrical methods, fault models that efficiently represent the effect of physical defects at some level of abstraction have been described in [18]. These models can be used to capture the effect of physical defects that produce incorrect behavior. Faults can be caused by manufacturing imperfections, or by degradation during use as electrodes are actuated. Possible causes of defects are listed below.

- *Dielectric breakdown:* High voltage levels during actuation cause dielectric breakdown, which creates a short between a droplet and the underlying electrode. In this case, droplet transportation cannot be controlled since the droplet undergoes electrolysis.
- *Degradation of the insulator:* The actuation of electrodes for long durations causes charge to be irreversibly concentrated near the electrodes. This impedes droplet transportation because of the undesired variation of interfacial surface tension along the droplet flow path.
- *Short-circuited electrodes:* A short between two adjacent electrodes effectively forms one longer electrode. When a droplet resides on this electrode, it is no longer large enough to overlap the gap between adjacent electrodes. As a result, the actuation of the droplet can no longer be achieved.
- *Open in the metal connection between the electrode and*

*This work was supported in part by the National Science Foundation under grant CNS-1135853.

the control source: This open can be on the path between the electrodes and the chip pads, or through the external wires between the biochip and the adjacent controller. This defect results in a failure in activating the electrode for droplet transport.

The correctness of bioassay outcomes can be determined by utilizing on-chip detectors. In addition, physical-aware control software can be used in a DMFB platform to implement an error-recovery method. Section II discusses the key parameters underlying effective error recovery; this discussion leads to the drawing of the error-recovery design space. The available biochip resources and the control system are considered as guidelines to identify appropriate error-recovery components. The process of designing error-recovery components is described in Section III. A real-life experiment for error recovery based on a biochip that was fabricated in our laboratory, coupled with a sensor circuit, is illustrated in Section IV. Section V concludes the paper.

II. Efficacy of error recovery in diagnostics-on-a-chip

To assist with verifiable bioassay execution in POC clinical diagnostic settings, researchers have recently presented a set of error-recovery schemes that are suitable for DOCs [17]–[20]. These schemes vary in their design characteristics with respect to resource utilization and response time. Although these proposals have not provided formal methods for design-space exploration, tradeoffs between error-recovery (response) time, area overhead, control-memory usage, and scalability can be inferred based on their methodologies. Accordingly, multidimensional design-space exploration is required in order to help system designers to pinpoint recoverability limitations, potentially using statistical analysis, for a given configuration.

We begin our design-space exploration by introducing some formal definitions that represent the various optimization knobs. Then, we evaluate the proposed methods in [17]–[20] in a fair manner using these definitions.

A. Responsiveness

At the time when an error occurs, the control software requires a period of time, called *recovery time* T_r, to recover from the error and ensure error-free execution. This time is a measure of system responsiveness; *i.e.*, the shorter the time, the better the responsiveness. Precisely, this period of time consists of two phases: 1) *Recovery-procedure triggering time* (T_g), which is the duration between error occurrence and error detection; 2) *Recovery-procedure execution time* (T_e), which is the time required by the procedure to ensure error-free execution from the same (previously erroneous) operation. Note that the system responsiveness R is inversely proportional to the recovery time.; *i.e.,* $T_r = T_g + T_e$ and $R \propto \frac{1}{T_r}$.

Typically, the triggering time T_g depends on the technology of the sensor attached to the system. Deploying a fluorescence detector such as in [19] or a capacitive sensor as in [18] forces the system designer to insert checkpoints to detect errors. During execution, droplets need to be driven into these locations for verification. Hence, it is possible that an error occurs at a certain time but it is not detected until the droplet is driven subsequently to a checkpoint, thereby making T_g relatively long. On the other hand, the use of a CCD camera-based sensing system leads to negligible triggering time since an error can be detected immediately, as shown in [17], [20].

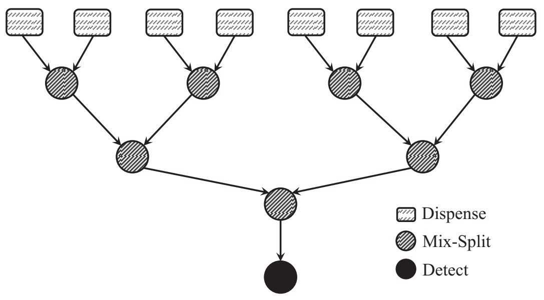

Fig. 1: A sequencing graph on PCR for DNA amplification.

The recovery-procedure execution time T_e is linked to the class of recovery system coupled with the chip. In [19], Zhao *et al.* provide control paths in which the insertion of detection checkpoints and the generation of copy droplets are embedded in the initial synthesis result. Therefore, T_e is relatively short. A shorter recovery execution time is seen in [20], where the recovery sequence is loaded directly from on-chip memory. The dynamic resynthesis procedure in [17], however, requires more time for dynamic adaptation, making T_e longer.

B. Area Overhead

A popular strategy adopted in fault-tolerant computing is spatial redundancy of hardware units to generate reliable outputs. We exploit this idea for error recovery in DOC. During bioassay execution, some backup droplets (also called copy droplets) are kept on the chip array to be used as starting points for the recovery sequence. Therefore, the larger the number of copy droplets, the shorter the recovery sequence. However, the presence of copy droplets at the boundary cells of the array imposes additional space constraints on regular bioassay operations. Therefore, a control system that executes an assay containing N operations (*i.e.*, nodes) and produces C copy droplets is said to have backup ratio BR such that $BR = \frac{C}{N}$.

Note that BR never reaches the value 1 since not all operations can produce copy droplets. For instance, the conventional PCR process shown in Fig. 1 (an essential part of any clinical diagnosis protocol) contains eight dispense nodes, seven mix-split nodes, and one detection node. The mix-splits are the only operations that can produce copy droplets. As a result, a control system that enables storing all possible copy droplets will have $BR = 7/16$. Note that producing more copy droplets does not necessarily mean shorter recovery time. Copy droplets can cause operations to stall due to space limitations. Therefore, other error-recovery systems store fewer droplets, resulting in smaller BR.

Another aspect of recovery-space cost can be analyzed on the basis of reliability. Errors such as the generation of droplets with abnormal volumes are usually caused by the accumulation of charge on the surface of certain electrodes [21]. If the use of such electrodes is continued, it is likely that they will introduce more errors. Thus, in order to ensure the reliability of DOC, the electrodes at which an error has been deemed to have occurred should be bypassed in the new synthesis results [17]. Note that the number of discarded cells due to the reliability requirement is tightly coupled with the type of sensor used in the system. For instance, the systems in [18], [19] use fluorescence detectors and capacitive sensors, respectively. In these systems, the faulty electrode cannot be precisely located. Therefore, *region-level* reliability is adopted by discarding a path (region) of cells that represent the path of the erroneous droplet. On the other hand, a CCD camera-based sensing system is used in [17], [20] which is able to accurately locate the faulty electrode; thus adopting *cell-level* reliability. We define the reliability cost to be the average number of discarded cells per recovery run.

C. Control-memory usage cost

Quantification of control-memory usage also depends on the class of recovery system used. Luo et al. in [17] employ an *a posteriori* approach in which a dynamic resynthesis technique is invoked to recover from an error. In this case, no prior recovery information needs to be stored; thus memory utilization is minimized. On the other hand, the work in [20] follows an *a priori* approach such that all the possible recovery sequences for the operations are stored in memory before actual execution. This class targets real-time applications that demand rapid response. Nevertheless, a significant amount of memory-storage is required to cover all error possibilities. There is clearly a tradeoff between the response speed and the memory utilization in error-recovery systems.

D. Recovery from multiple concurrent errors

Handling situations when multiple errors occur concurrently is not a trivial problem. Recovering from multiple errors requires not only additional support from the sensing system, but also support from the recovery controller to prioritize recovery sequences [17]. For example, the error recovery scheme in [19] is unable to handle these situations, not because of the fluorescence detectors, but because the offline synthesis can only handle a recovery sequence from one error at a time. In case the fluorescence detectors report the occurrence of multiple errors, the recovery from these errors is performed sequentially. This strategy leads to inefficiencies in the recovery system. On the other hand, the cyberphysical approach in [17], either using fluorescence detectors or CCD camera-based sensors, is able to dynamically resynthesize the bioassay sequence upon error detection. In this situation, multiple recovery processes are triggered at the same time and the control software generates a priority queue for each recovery process. After these priority queues are merged, the control software assigns a priority for each element based on topological sort. From a practical point of view, this feature cannot be easily supported in [20] due to the exponential growth of the amount of memory required to store all possible combinations of errors.

Based on the four parameters defined above, we are able to develop a profile of the parameter space for error-recovery. Based on this reverse-engineering process, we can export this knowledge into a system design engine that selects an appropriate error-recovery scheme for a given configuration.

III. The art of error recovery design

Limitations in technologies related to CPU processing, memory, biosensor technique, and actuation capability impose constraints on error-recovery capabilities. These limitations motivate careful design in order to fully exploit the parameter space described in the previous section. In this section, we give some examples of how decisions can be taken for designing an error-recovery system that fits a certain configuration.

A. Simple memory-limited error recovery for a large chip

1) Problem Statement: We are given a large array that is intended to be used for rapid testing based on a protocol. The chip array is coupled with a standard optical sensor such as a fluorescence detector. The objective is to design a simple error-recovery scheme that uses limited memory storage.

2) Solution: To provide the targeted DOC with a suitable error-recovery scheme, we map the aforementioned system attributes into the parameter space. Since the system is equipped with a typical fluorescence detector, we have to incorporate a set of monitoring checkpoints at which verification is performed [19]. Undoubtedly, the system responsiveness in erroreous cases will be adversely affected due to the recovery-procedure triggering time T_g.

With this sensor technology in hand, we have two choices for developing a recovery controller that operates based on the incorporated checkpoints. The first choice is to store the recovery sequences for all the checkpoints in memory. Then, a sequence will be immediately loaded if needed (*i.e.,* on error detection). Note that this controller does not interrupt any of the other ongoing bioassay-related fluidic operations. However, this choice requires a large memory, which is not available in this setting. As a result, instead of storing all recovery sequences in advance, the second choice is to implement a *rollback* recovery that re-executes a portion of the protocol sequence. This rollback approach can be developed by empowering the controller with a dynamic resynthesis ability that produces new resource bindings, module placements, and droplet routings whenever an error is detected [17]. However, resynthesis execution is time-consuming because every component in the synthesis toolchain needs to be invoked. An alternative rollback methodology that does not perform resynthesis, but makes use of the available chip area, is to have a sufficiently large number of copy droplets that can be flexibly transported on-demand and used if needed. This principle is adopted in [19].

The proposed concept of rollback recovery redefines a checkpoint to be a monitoring as well as a copy-droplet storage location. Therefore, if an error is detected at a checkpoint, the rollback procedure will re-execute all fluidic operations from the immediate upstream checkpoint along the paths in the protocol sequence. The re-execution procedure will benefit from the copy droplet stored at this upstream checkpoint. Note that a significant portion of the chip area is utilized now for storing copy droplets for the checkpoints. This cost is, however, justifiable given the large chip size.

B. *Delay-tolerant memory-aware error recovery for a practical chip*

1) Problem Statement: From a practical point of view, the array size of the given DOC is small. Therefore, we have a setting where we are not able to keep many copy droplets on the chip. In some cases, a checkpoint will be used for monitoring without the storage of any copy droplets. Thus, the rollback procedure demands more intelligence in deciding which path of operations needs to be re-executed. Moreover, the on-chip memory storage is also limited.

Fortunately, our system in this scenario is provided with a CCD camera-based sensor. Hence, the objective is to leverage the benefit of CCD camera monitoring to architect an error-recovery scheme that can dynamically respond to errors by considering the memory-storage limitation.

2) Solution: Since the chip is equipped with a CCD camera-based sensor, we no longer need to determine the locations of checkpoints to be inserted during initial design synthesis. The CCD camera can be used in experiments to show the plan view of currently moving droplets [17], as shown in Fig. 2(a). Based on the images captured by the CCD camera, droplets can be automatically located by the control software using image processing. The control software generates a correlation map between the array image and a pattern that represents the image of a typical droplet [17], [22]. Fig. 2 shows an image that is captured by the CCD camera as well as the generated correlation map. Obviously, this approach not only enables the control software to detect errors immediately and therefore eliminating the triggering time T_g, but it is also capable of precisely identifying the cell location of the error; *i.e.,* providing cell-level error localization. As a result, the CCD camera can be efficiently used to observe the cyberphysical system. This advantage justifies the additional cost incurred due to the additional sensor instrumentation.

With the availability of on-chip sensors and the hardware that can send feedback to the control software, it is now necessary to design physical-aware software that can analyze sensor data and dynamically adapt to it. Adaptations includes updates for the schedule of fluid-handling operations, resource binding, module placement, and droplet routing pathways. Therefore, based on the work proposed in [17], the task of the recovery controller includes the following two phases: 1) The pre-execution (offline) data preparation phase, which is essentially the initial synthesis of the protocol sequencing graph; 2) The online monitoring and dynamic adaptation phase,

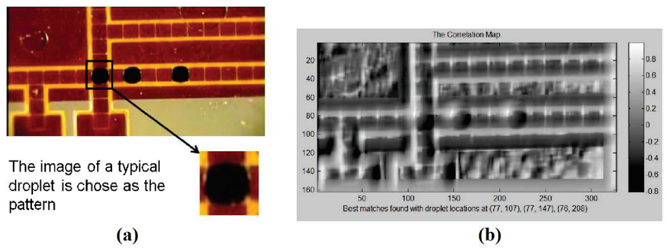
The image of a typical droplet is chose as the pattern
(a)
The Correlation Map
20
40
60
80
100
120
140
160
50
100
150
200
250
300
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
Best matches found with droplet locations at (77, 107), (77, 147), (76, 208)
(b)

Fig. 2: Illustration of CCD camera-based monitoring system. (a) Captured image. (b) Correlation map between the array image and the pattern [17].

which provides an algorithmic resynthesis capability for the controller [17]. Hence, during the execution of the bioassay, detection of an error will trigger error recovery; a backtrace will be performed in the original sequencing graph to identify which portions of the previously executed operations need to be re-executed. In this approach, the backtracing path depends on how many copy droplets are available on the chip.

A drawback of this cyberphysical error-recovery scheme is the delay incurred in adaptation due to the need for resynthesis. Such a resynthesis step also causes other fluid-handling operations to be interrupted. However, compared with the timing of the previous scenario, a small increase in T_e is balanced with the negligible value of T_g. Moreover, the setting here is delay-tolerant. Finally, the on-chip memory usage is relatively larger than for the previous scenario since both the original and updated synthesis results need to be saved. Moreover, data resulting from image-processing steps needs to be handled. Nevertheless, no significant memory storage is needed since the stored resynthesis and image processing data are continually updated during runtime.

C. *Real-time (fast) error recovery for a practical chip*

1) Problem Statement: Several clinical diagnosis settings demand that the reactions are rapidly carried out and carefully controlled in real-time to produce desired compounds with high selectivity [20], [23]. This requires highly precise time-control in each step of chemical synthesis. In this third scenario, we are equipped with a chip that has large on-chip memory coupled with a CCD camera-based sensor that is able to monitor the status of moving droplets in real-time. The objective is to take the advantage of the large memory to develop a fast error-recovery system; *i.e,* with negligible recovery time T_r.

2) Solution: Based on the above analysis, we note that it is time-consuming to dynamically resynthesize the chip protocol whenever an error is detected. To support real-time error recovery, we need to pre-compute and store recovery sequences for all errors of interest that can occur during a bioassay. This issue is highlighted in [20]. When an error is detected by on-chip sensors during the execution of a bioassay, the controller can simply look up the recovery solution (known as *dictionary element*) in memory rather

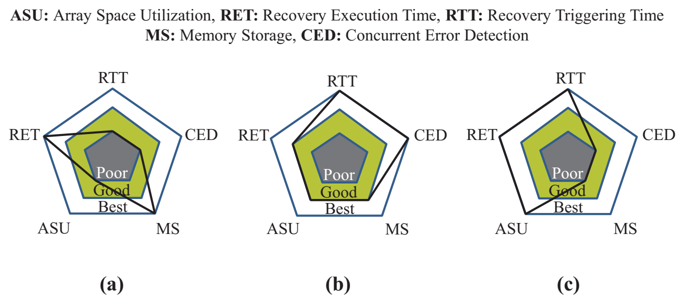

Fig. 3: Illustration of the tradeoffs apparent in the following recovery systems: (a) checkpoint-based offline error recovery [19], (b) software-based cyberphysical error recovery [17], and (c) dictionary-based real-time cyberphysical error recovery [20].

than performing online resynthesis. This dictionary-based solution therefore reduces response time. The generation of these dictionary elements can be performed using simulation before the execution of the experiment. Also, to minimize the memory required in this hardware-assisted error-recovery method, a fast compaction/de-compaction technique can be coupled with the finite-state machine (FSM). The compaction and de-compaction procedures do not impact the response time, as shown in [20].

Despite the significantly low response time, the number of generated dictionary elements increases dramatically if multiple-operation error recovery is considered. Therefore, such a system requires substantial on-chip memory storage. Also, the availability of copy droplets at specific locations must be communicated to the recovery controller in advance such that the droplet can be easily transported and utilized. This information also provides the system with full controllability over the entire chip array.

D. Qualitative Comparison

Fig. 3 qualitatively summarizes the tradeoffs inherent in the recovery systems described in the previous sub-sections. The first platform discussed in Section III-A is referred to as *checkpoint-based offline error recovery*. The DOC platform illustrated in Section III-B is referred to as *software-based cyberphysical error recovery*. Finally, the fast error-recovery dicussed in Section III-C is called *dictionary-based real-time cyberphysical error recovery*.

A key observation is that recovery support for multiple concurrent errors is tightly coupled with the type of on-chip sensor technology. A controller that receives feedback signals from checkpoint-based fluorescence detector is not able to directly support this capability. A CCD camera-based sensor, on the other hand, facilitates execution of such fine-grained monitoring. However, in dictionary-based error recovery, although the CCD camera-based sensor is available, the large number of error combinations that need to be considered *a priori* significantly complicates concurrent multiple-operation error recovery.

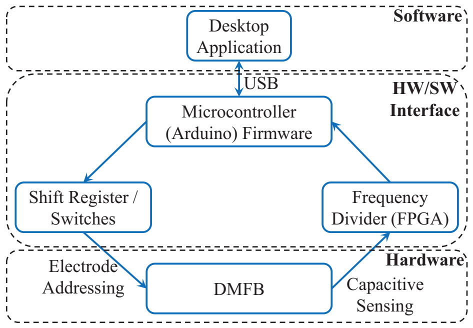

Fig. 4: The setup of the control system loop [18].

IV. Experimental Case Study

As a case study of error-recovery realization, we discuss the recent implementation of a hardware-assisted error recovery platform [18]. We describe an integrated demonstration of cyberphysical coupling in DOC, whereby errors in droplet transportation on the chip array are detected using capacitive sensors. We also show how the test outcome is interpreted by the controller to autonomously accomplish error recovery as needed.

A. Control system setup

The physical setup for error detection and recovery demonstrates the coordination between the hardware (chip and sensors) and the control software. The hardware/software interface is realized through seamless interaction between control software, an off-the-shelf micro-controller, a shift register for synchronous actuation of 32 chip electrodes, and a frequency divider implemented on an FPGA [18]. The schematic of the control system is shown in Fig. 4. The intermediate micro-controller provides the following functionalities: 1) communication with the desktop application via a USB link, 2) a parallel-to-serial conversion for the 32-bit electrode actuation vector provided by the desktop and injecting the serial data into the shift register, 3) metering the signal frequency received from the FPGA to interpret the capacitive sensing readout. The details of the sensing circuit will be explained in the following subsection.

On the hardware side, some electrodes are equipped with the sensing circuit; thus designated as *checkpoints*. For effective error recovery, a number of such checkpoints must be incorporated [18]. However, precise localization of errors is not possible with this sensor technology because the only information available to us is that a droplet is stuck within a region between two checkpoints. Therefore, to ensure reliability whenever an error is detected, a region-level bypassing is employed during control software rollback to recover from the error. Based on this scenario, the chip array is divided into several regions with a checkpoint associated with each region. When a "missing droplet" error is detected at a checkpoint, it can be inferred that the defect lies on the path between this checkpoint and the previous checkpoint on the designated droplet route, thus rollback is initiated. A droplet under test retraces its path from the current checkpoint to the previous checkpoint. Note that the rollback is deemed to have been successful when the previous checkpoint once again reports the presence of a droplet (after a known number of clock cycles). If retracing fails, *i.e.,* the previous checkpoint does not report a droplet, we conclude that the droplet is irreversibly stuck [18]. As a result, a new droplet needs to be dispensed and the cells between the two checkpoints have to be permanently discarded for reliability.

B. *Capacitive sensor technology*

To enable the detection of nanoliter scale droplets on a DOC platform, a capacitance sensing circuit was designed in [18]. The sensor relies on a ring oscillator to test droplet presence (instead of testing droplet volume, which might require a considerable amount of manual intervention). This is based on the inverse relationship between droplet volume V and ring oscillator frequency f; *i.e.,* $V \propto \frac{1}{f}$. It also depends on the ability of a high-speed frequency measurement unit and comparator to classify signals corresponding to the presence or the absence of a droplet [18]. The oscillator circuit is used to monitor the capacitance between the control and ground electrodes in the chip actuator giving an output that is a 0-3.3 V frequency-decoded square wave, ranging from 750 kHz to 1.5 MHz, with a 50% duty cycle [18].

C. *Diagnostics-on-a-chip specification and layout*

A 32-electrode chip was divided into four regions. The positions of checkpoints are determined to ensure that each region has exactly one checkpoint. From the pre-computed activation sequences, the arrival time for a droplet at a checkpoint is known. Hence, based on the sensor readout, the control software decides whether a droplet has arrived as expected. If a droplet reaches a checkpoint as planned, the experiment will be continued and the droplet will enter the next region. If not, the reconfiguration steps will be triggered automatically, a backup route will be generated in real-time and the new electrode activation sequences will be immediately fed into the biochip.

D. *Results and video*

Several experimental runs were carried out using the fabricated chip, sensing hardware, the hardware/software interface, and the control software described before. In order to capture a video for the experiment, the researchers in [18] used a CCD camera. Videos illustrate successful re-routing of droplets and bypassing of faults whenever an error is detected.

V. Conclusion

We have described recent proposals for error recovery in a DMBF, which is key for clinical diagnostics applications. We have described four evaluation parameters that form the error-recovery design space. Based on these evaluations, the design of an efficient error-recovery scheme has been outlined for a given a specific system configuration. As a case study, we have described an experimental setup for a control system that was previously used to provide real-time error recovery for DMFBs.

References

[1] R. Peeling and D. Mabey, "Point-of-care tests for diagnosing infections in the developing world," *Clinical Microbiology and Infection*, vol. 16, no. 8, pp. 1062–1069, 2010.

[2] N. P. Pai *et al.*, "Point-of-care testing for infectious diseases: diversity, complexity, and barriers in low-and middle-income countries," *PLoS Medicine*, vol. 9, no. 9, p. e1001306, 2012.

[3] N. Pant Pai and M. B. Klein, "Are we ready for home-based, self-testing for HIV?" *Future HIV Therapy*, vol. 2, no. 6, pp. 515–520, 2008.

[4] K. Lewandrowski *et al.*, "Implementation of point-of-care rapid urine testing for drugs of abuse in the emergency department of an academic medical center: Impact on test utilization and ED length of stay," *American Journal of Clinical Pathology*, vol. 129, no. 5, pp. 796–801, 2008.

[5] C. H. Ahn *et al.*, "Disposable smart lab on a chip for point-of-care clinical diagnostics," *Proc. of the IEEE*, vol. 92, no. 1, pp. 154–173, 2004.

[6] R. Sista *et al.*, "Development of a digital microfluidic platform for point of care testing," *Lab on a Chip*, vol. 8, no. 12, pp. 2091–2104, 2008.

[7] Y. Zhao *et al.*, "Droplet manipulation and microparticle sampling on perforated microfilter membranes," *Journal of Micromechanics and Microengineering*, vol. 18, no. 2, p. 025030, 2008.

[8] R. B. Fair, "Digital microfluidics: is a true lab-on-a-chip possible?" *Microfluidics and Nanofluidics*, vol. 3, no. 3, pp. 245–281, 2007.

[9] K. Chakrabarty, "Design automation and test solutions for digital microfluidic biochips," *IEEE Trans. Circuits Syst. I*, vol. 57, no. 1, pp. 4–17, 2010.

[10] T.-W. Huang *et al.*, "Integrated fluidic-chip co-design methodology for digital microfluidic biochips," in *Proc. ISPD*, 2012, pp. 49–56.

[11] F. Su and K. Chakrabarty, "Architectural-level synthesis of digital microfluidics-based biochips," in *Proc. ICCAD*, 2004, pp. 223–228.

[12] ——, "Unified high-level synthesis and module placement for defect-tolerant microfluidic biochips," in *Proc. DAC*, 2005, pp. 825–830.

[13] F. Su, W. Hwang, and K. Chakrabarty, "Droplet routing in the synthesis of digital microfluidic biochips," in *Proc. DATE*, vol. 1, 2006, pp. 1–6.

[14] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang, "BioRoute: A network-flow-based routing algorithm for the synthesis of digital microfluidic biochips," *IEEE Trans. CAD*, vol. 27, no. 11, pp. 1928–1941, Nov 2008.

[15] T.-W. Huang and T.-Y. Ho, "A two-stage integer linear programming-based droplet routing algorithm for pin-constrained digital microfluidic biochips," *IEEE Trans. CAD*, vol. 30, no. 2, pp. 215–228, Feb 2011.

[16] Z. Xiao and E. Young, "CrossRouter: A droplet router for cross-referencing digital microfluidic biochips," in *Proc. ASP-DAC*, 2010, pp. 269–274.

[17] Y. Luo, K. Chakrabarty, and T.-Y. Ho, "Error recovery in cyberphysical digital microfluidic biochips," *IEEE Trans. CAD*, vol. 32, no. 1, pp. 59–72, Jan 2013.

[18] K. Hu *et al.*, "Fault detection, real-time error recovery, and experimental demonstration for digital microfluidic biochips," in *Proc. DATE*, 2013, pp. 559–564.

[19] Y. Zhao, T. Xu, and K. Chakrabarty, "Integrated control-path design and error recovery in the synthesis of digital microfluidic lab-on-chip," *J. Emerg. Technol. Comput. Syst.*, vol. 6, pp. 11:1–11:28, 2010.

[20] Y. Luo, K. Chakrabarty, and T.-Y. Ho, "Real-time error recovery in cyberphysical digital-microfluidic biochips using a compact dictionary," *IEEE Trans. CAD*, vol. 32, no. 12, pp. 1839–1852, Dec 2013.

[21] H. Verheijen and M. Prins, "Reversible electrowetting and trapping of charge: model and experiments," *Langmuir*, vol. 15, no. 20, pp. 6616–6620, 1999.

[22] Y.-J. Shin and J.-B. Lee, "Machine vision for digital microfluidics," *Review of Scientific Instruments*, vol. 81, no. 1, p. 014302, 2010.

[23] J.-I. Yoshida, "Flash chemistry: flow microreactor synthesis based on high-resolution reaction time control," *The Chemical Record*, vol. 10, no. 5, pp. 332–341, 2010.