Parallel Software Design Enabling High-Speed Reliability Testing of Inkjet Printheads

George Ungureanu[1,2], Ingo Reinhold[1,2], Ingo Sander[2], Werner Zapka[1,2]; [1]XaarJet AB, Järfälla, Sweden, [2]KTH Royal Institute of Technology, School of ICT, Stocholm, Sweden

Abstract

With new functional applications emerging in the digital printing industry, the need for quantitative knowledge of the reliability of drop-on-demand inkjet printheads increases. Continuous ink circulation using TF Technology™ and the resulting channel self-recovery is one of the technologies which decrease the down-time of a single nozzle, but in turn increase the difficulty of an accurate reliability test. Current measuring techniques, namely the a-posteriori verification of printouts on paper proved to be inappropriate.

This paper proposes a novel software approach, exploiting signal processing techniques, strong control loops and powerful system design methodologies in order to allow for the correct detection of single missing droplets at run-time. This new system is meant to relieve the effects of the indefinite environment and sources of human error. Preliminary results and the proof-of-concept demonstrates both the system's and the design method's versatility and potential.

Introduction

Drop-on-Demand inkjet printing [1] is a versatile technology that can be used on most substrates and with a wide variety of inks and dyes, based on water, solvent or oil. TF Technology™ implies printheads with a continuous ink flow through channels, that is able to remove the particles or air ingestions, enabling self-recovery during printing. Both the high-throughput and the management of picoliter-sized droplets make this technology suitable for media printing as well as for functional printing. Functional applications include printing of RFID antennas, solar cells or micro array lenses. These features put high challenges on the achievement of reliability tests.

In this contribution we optimize a recently developed system which combines advanced optics with an efficient software algorithm to capture droplets in-flight at full jetting frequency and determine off-line the functional state of the printhead for every droplet ejected. In this fashion the cumulative influence of ink dynamics, cross-talk, nozzle plate flooding and the printed image are condensed qualitatively into the statistics of missing droplets over one full actuator. The used image segmentation algorithms however, proved to be prone to image artifacts originating from, for example, misfiring neighboring nozzles. Although sufficient for detecting errors relevant to graphical applications, this system is far from optimal in what concerns functional applications, where detecting single-drop events may be crucial.

The functionality of the proposed component combines digital signal processing with control loops in order to overcome the two main factors that reduce its reliability: the work environment hazards and the human errors.

The experimental setup

The experimental setup consists of two parts: the optical rig and the analysis component. The former is a hardware setup meant to capture images of droplets in-flight using a line scan camera [2]. The latter is a software component that analyzes the captured images and plots quantitative information that describes the printhead's functioning state. Initially, the analysis was meant to be on-line, using a system with general purpose graphical processing units (GPGPUs) [2]. Due to the software's instability however, it was off-loaded into an off-line version which employs a desktop workstation [3]. The data processing is done a-posteriori on images recorded at run-time.

The optical rig

The experimental hardware setup is shown in Figure 1. It consists of four main parts: a light source (1), a printhead (2), an ink catcher (3) and a line scan camera (4).

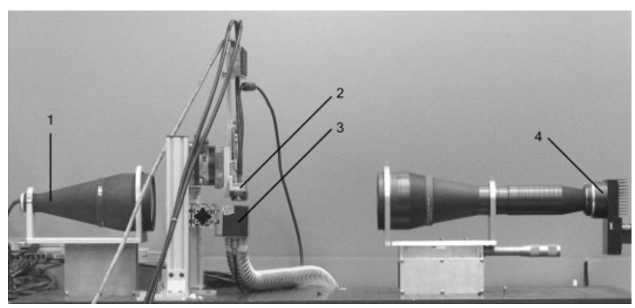


Figure 1. *The optical rig*

The light source emits parallel light that enables the released droplets to cast high-contrast shadows upon the line scan camera. The printhead (Xaar 1001, Xaar, UK) is printing a known pattern which is used for comparison by the analysis process. The ink catcher collects the ink ejected by the printhead. The line scan camera (Pirahna 3, Dalsa, CAN) is a CCD one-dimensional sensor array that catches droplet shadows and sends 140 MB/s of data to the analysis component.

The software component

The software analysis is currently done in three steps: image recording, tickmark detection, result plotting.

A sample from a recorded image is shown in Figure 2. It depicts the droplet shadows printing a known pattern (the shape of digit 4) in three cycle mode. The linear shapes represent droplet shadows distributed in space (horizontal axis) depending on the originating nozzle, and in time (vertical axis) corresponding to the sample captured by the line scan camera.

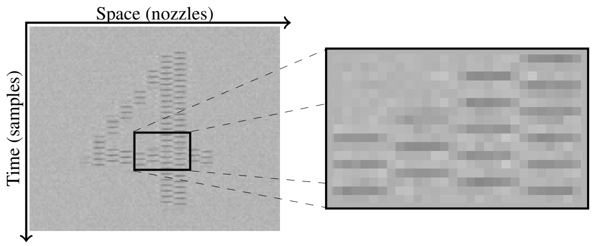


Figure 2. *Zoomed image sample captured by the line scan camera, showing digit 4*

The tickmark detection step is done through a segmentation algorithm which calculates the exact position of the areas of interest (AOI) where the drops should be positioned, and then applying a median filter [4] upon the AOI. A binarization process detects the existence of droplets within their AOI, based on a user-determined threshold, as seen in Figure 3. A series of undetected droplets implies that the nozzle is malfunctioning and the event is recorded as a *tickmark*.

The detection results are gathered and used in the third step of the software analysis, where run-time information is plotted in the form of quantitative statistics about the printhead's reliability status, as shown in Figure 4.

Challenges

Although the presented printhead reliability system is a robust implementation that provides automatic analysis reports far superior to the a-posteriori verification of printouts on paper, there is a number of external factors that prevent it from being fully reliable for validating printheads targeting functional applications. Most of these unreliabilities originate from ambiguous tickmarks that cannot be validated due to the unstable capturing environment or due to human errors. The following paragraphs will present the three main challenges regarding tickmark analysis, which are treated in this paper.

Improper image capturing conditions are caused by hardware or environment noise, or an improperly calibrated setup. Two examples are shown in Figure 5. As can be seen, in such conditions the user-set threshold becomes ineffective during the binarization process, rendering an unreliable tickmark detection.

A high potential for human errors is determined by the numerous parameters that need to be manually set up during several iterative and inter-dependent steps. The two most common situations are depicted in Figure 6 where the initial parameters lead to improper binarization results or incorrectly calculated AOIs.

Optical artifacts are unknown objects that obstruct the camera's line of sight and are the most common source of errors in the present system. As can be seen in Figure 7, the captured artifact is interpreted as a droplet where in fact the nozzle is malfunctioning.

Other encountered problems include mechanical influences, the improper detection of droplets with dynamic velocity, non-constant background lighting, etc. The current system ignores tickmarks smaller than 100 droplets since their existence is ambiguous, and they are considered too small for the human eye to recognize.

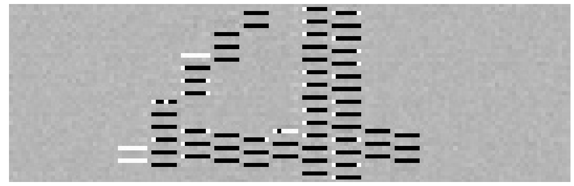
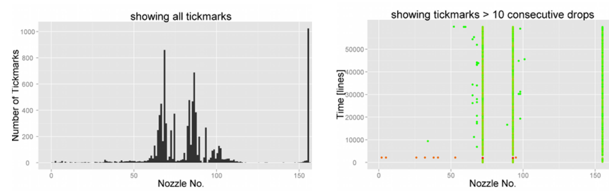
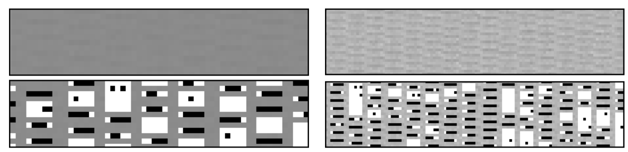


Figure 3. *Zoomed superposition of initial image and binarization result. Black = detected droplet shadow; White = undetected droplet shadow*



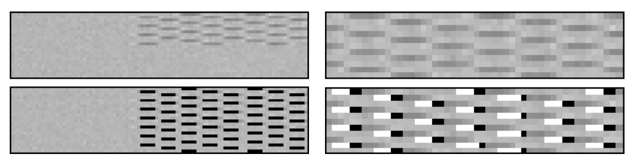
(a) Total number of tickmarks (b) Tickmark occurence in time

Figure 4. *Examples of result plots evaluated for each nozzle after total occurrence (a) and occurrence in time (b)*



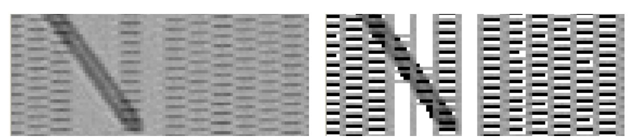
(a) Insufficient contrast (b) Low signal-to-noise ratio

Figure 5. *Examples of improper capturing conditions. Comparison between the input images (top) and the binarization results (bottom)*



(a) Incorrect threshold (b) Misalligned position parameters

Figure 6. *Examples of human inflicted errors. Comparison between the input images (top) and the binarization results (bottom)*



(a) Input image (b) Binarization result

Figure 7. *Examples of optical artifact*

Optimization approaches

The current software system's strong sequential implementation and its iterative work stages impede its translation into an on-line solution for printhead reliability. Therefore we propose a different design approach aimed at:

- boosting the throughput through a parallel design methodology aimed at implementing on heterogeneous hardware architectures that contain massively parallel platforms;
- minimizing the rate of human errors through adaptive calibration and control processes;

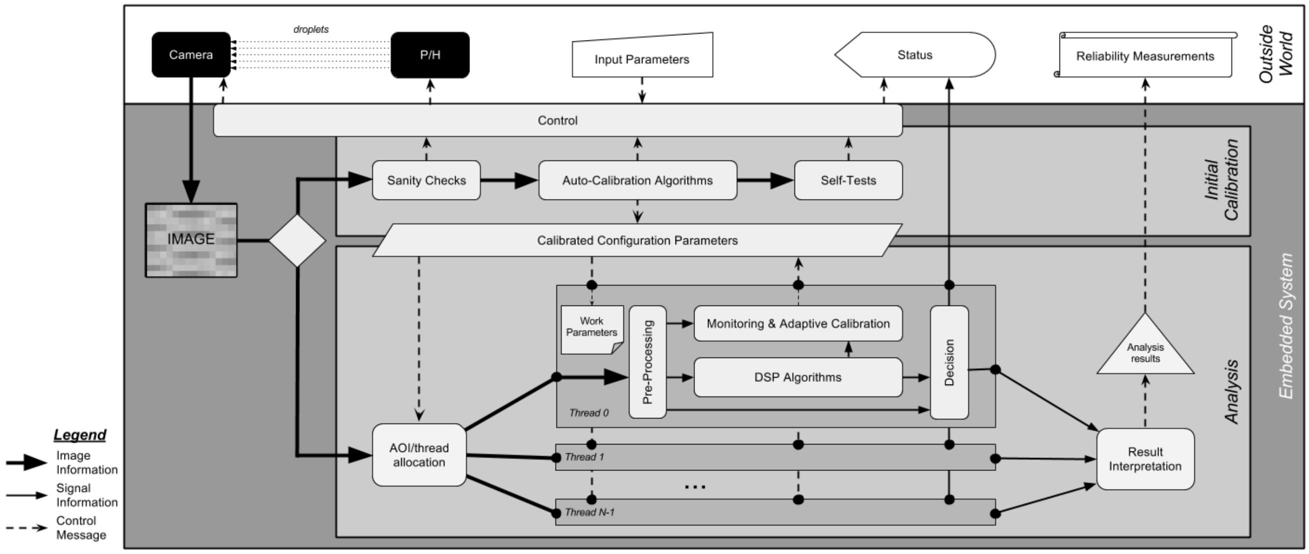


Figure 8. *The overall block diagram for the proposed printhead reliability system*

- increasing the analysis reliability through image and signal processing techniques targeting parallel platforms.

An overall block diagram of the cyber-physical system [5] for reliability testing of inkjet printheads is presented in Figure 8. The graphic depicts the interactions between an "outside world" (rig, interfaces) and the embedded system, which performs the analyses. The latter can be in either an *initial calibration* state or a *run-time analysis* state.

An initial calibration is performed every time the system starts or decides to invalidate the analyses in case of extreme environment changes. During this stage the hardware components and the software parameters are adjusted through repeated control loops.

The run-time analysis starts after the system is optimally calibrated. Based on the parameters extracted in the previous stage it assigns sections of the input image to different processing resources to be refined in parallel. Afterwards it gathers the results for interpretation and plotting. Each parallel resource is represented in Figure 8 as a *thread*, which handles data originating from one printhead nozzle. During the analysis process, the image is transformed into a one-dimensional (1D) control signal, and is subjected to a series of digital signal processing (DSP) algorithms to extract useful information from the noisy images. At the same time a process monitors all changes in the environment and dynamically adapts the working parameters for each thread accordingly.

The following sections will present three of the main algorithms employed by the system depicted in Figure 8.

Nozzle Auto-Calibration

This algorithm is part of the initial calibration stage and its purpose is to find the exact position of the individual printhead nozzles and set their AOIs. To achieve this a series of black and white stripes are printed and have to be detected by the software component which is equally distributed to parallel computing resources. The algorithm consists in the following steps:

1. *Background sanity check* is performed while the printer is off, and verifies whether the capturing conditions are proper for image analysis (the hardware is correctly set up and the environment does not suffer extreme changes). The software component is doing a vertical and a horizontal swipe of the captured image applying Equation 1 to assure that the white level stays within an accepted range, and acts accordingly.

$$|in[n] - \|in\|| < \sigma_{in} \cdot k_{\sigma}, \tag{1}$$

where $in[n]$ is the input sample, σ is the standard deviation [6] and k_{σ} is a deviation factor.
2. *First stripe detection* forces the software component to identify the first white-to-black transition when the printhead starts printing the first black stripe. Equation 2 is satisfied.

$$|sum - \|sum\|| > \sigma_{sum} \cdot k_{\sigma}, \text{ where } sum = \sum in[n] \tag{2}$$

3. *Printhead width detection and coarse calibration* identifies the printed area in the input image and equally distributes it to parallel threads as AOIs.
4. *Break and second stripe detection* implies that the printhed stops printing and additional background information is gathered for the newly allocated threads. The second line detection employs a faster version of Equation 2.
5. *Fine calibration* is done while the printhead prints the second stripe. Each thread accumulates a shadow density and filters out the irregularities. The position parameters align themselves according to these densities and the neighboring nozzles. The method takes care of the malfunctioning nozzles by analyzing the context and predicting where they should be positioned.

This algorithm results in a set of optimal position parameters that describe the AOI placement in the input image. It is followed by other *Auto-Calibration Algorithms* such as the adjustment of noise levels, drop per dot (DPD) levels, drop detection parameters, etc.

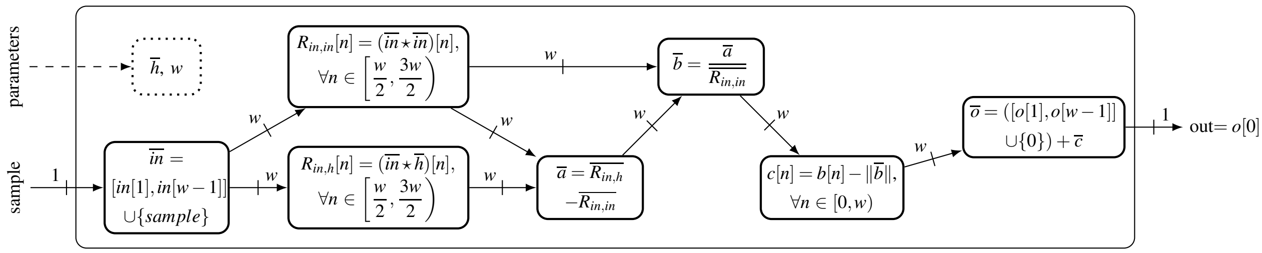


Figure 9. *Block diagram for the drop detection filter*

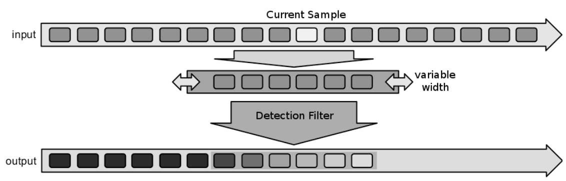


Figure 10. *The working principle for the drop detection filter*

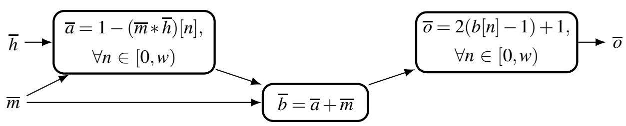


Figure 11. *Block diagram for the correction mask calculation*

Drop Detection

The drop detection is done through a set of DSP techniques that are applied upon a 1D signal associated with each nozzle. The main DSP algorithm is a drop detection filter which amplifies droplet information while reducing the effects of noise, artifacts or other anomalies. Its design is presented in Figure 9.

The filter inputs a stream of signal samples created from the information belonging to one printhead nozzle, as shown in Figures 12(a) and 12(b). It creates a vector (or *window*) of w samples from the input stream as suggested by Figure 10, where w is a configuration parameter. This vector is subjected to several transformations.

The main DSP operation employed is the *correlation* [6]. The *cross-correlation* ($R_{in,h}$) between the input vector $\overline{in}$ and an "ideal" window $\overline{h}$ determines the similarity of the input signal with a desired signal originating from a perfect droplet image (already binarized). Since according to Nyquist's sampling theorem [6], the input signal is regarded as noise, the *auto-correlation* ($R_{in,in}$) would describe the input's noise characteristic. Therefore by subtracting it from $R_{in,h}$ and normalizing the result against it, we would have a correct description of the droplets' behavior ($\overline{b}$). Since we are interested only in the peaks that describe droplets, adjusting the resulting signal with an offset equal to its power ($\overline{c}$) reduces the background noise level to negligible values.

The filter outputs a stream of samples which is extracted from the vector $\overline{o}$ as suggested in Figure 10. An example output signal is shown in Figure 12(c).

Since the filter's sensitivity determines the response time for abrupt changes, vital information about the printing status can be lost, as seen in Figure 12(c). Therefore, a set of correction coefficients are applied upon the filter's output signal. These coefficients are calculated using the algorithm in Figure 11, which inputs the correlation window $\overline{h}$ and the printing pattern $\overline{m}$. An example can be seen in Figure 12, where the signal in 12(c) is corrected with the coefficients in 12(d) and results in the output signal in 12(e).

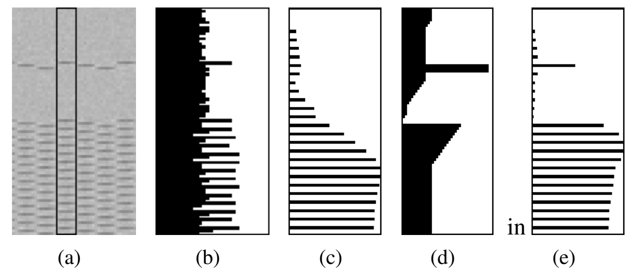


Figure 12. *Intermediate steps in the drop detection algorithm*

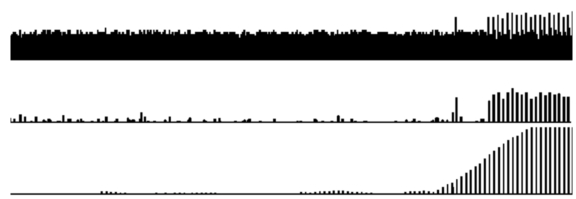


Figure 13. *The effect of different window sizes, assuming ideal environment and no correction mask: input signal (top);* $w = 6$ *(middle);* $w = 90$ *(bottom)*

Designing the drop detection filter implies a trade-off which has to be constantly monitored, as suggested in Figure 13. Increasing the vector width w would result in a cleaner and more stable output signal, but with a lower sensitivity and more computing resource demands. On the other hand, a lower w implies an unstable signal in hazardous environments, but a faster response to changes and lower resource demands.

Image pre-filtering

In order to increase both the system's reliability and its performance, an additional block is provided during the Pre-Processing stage: a small-footprint image filter. It is applied upon the 2D image streams allocated for each nozzle, before turning them into 1D signal data. The filter performs a

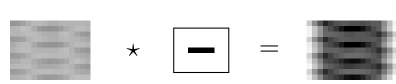


Figure 14. *The lightweight image pre-processing filter*

cross-correlation between a thread's AOI and an "ideal" droplet image, as shown in Figure 14. Since most of the artifacts and the environment effects are filtered out, the drop detection filter's input information will permit a lower w, while still maintaining a high degree of reliability.

Results

The three algorithms presented in this contribution have been individually tested using worst-case scenarios. The following paragraphs will show a few test samples to demonstrate the potential of these methods in particular and the printhead reliability cyber-physical system as a whole.

Nozzle Auto-Calibration

Figures 15 and 16 present two scenarios to demonstrate the usage of the nozzle auto-calibration algorithm. In both cases 32 nozzles were firing and captured in different areas of the input image. While Figure 15 shows the algorithm's functionality in normal (ideal) capturing conditions, Figure 16 demonstrates its robustness in imperfect environments.

Drop Detection

In order to demonstrate the robustness of the drop detection algorithm, a set of worst-case examples have been selected and are presented in this section.

In Figure 17 the result of an image originating from a complete printhead actuator filtered with the drop detection algorithm (17(c)) is compared against the input image (17(a)) and the results of the previous segmentation algorithm (17(b)). The input image is captured in a non-ideal environment displaying numerous optical artifacts which are successfully excluded.

Figure 18 presents several similar examples, scaled to observe individual events. It can be seen that the artifacts are correctly filtered out and the algorithm gathers as much relevant information as possible from the image context in order to rebuild droplet information. Therefore the droplets' existence is not ambiguous anymore and one can be sure that a detected tickmark is a real one. When analyzing the drop detection results, only lines which are completely black can be considered droplets, while gray-scale elements are ignored by the decision blocks.

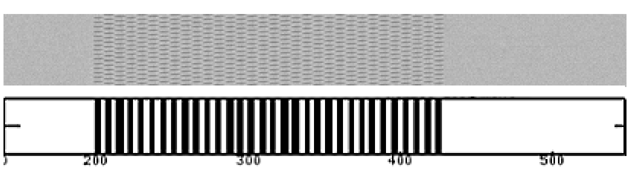


Figure 15. Example of AOI recognition in ideal conditions: input droplets image (top); identified AOIs (bottom)

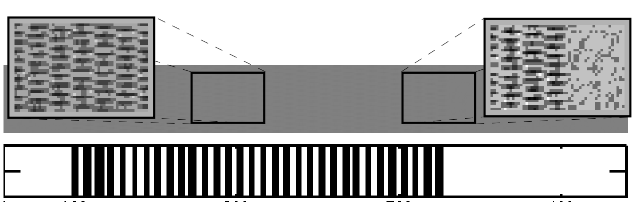
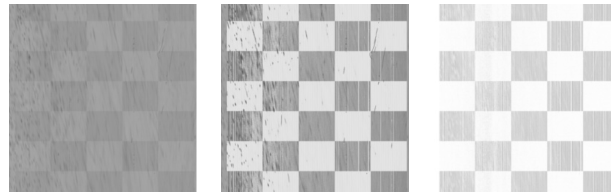
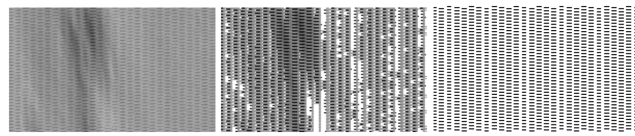


Figure 16. Example of AOI recognition in a bad contrast environment: input droplets image (top); identified AOIs (bottom); areas with artificially enhanced contrast (insets)

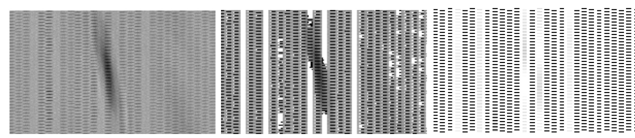


(a) Input image (b) Segmentation (c) Drop detection

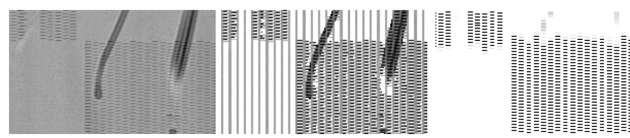
Figure 17. Overall image of 500 nozzles printing a checkerboard pattern



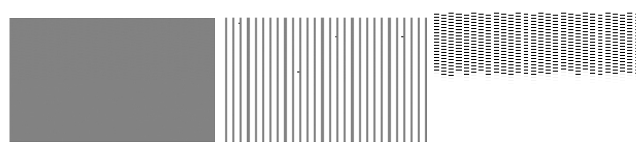
(a) Artifacts over droplets



(b) Artifact over tickmarks and droplets



(c) Artifact over tickmarks and droplets



(d) Capturing conditions with insufficient contrast

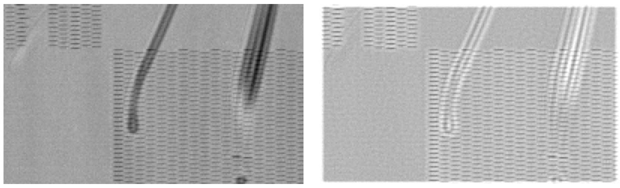
Figure 18. Zoomed examples of droplet detection, without correction mask: input image (left); segmentation results (middle); detection results (right)

Figure 19. Zoomed examples of droplet detection: input image (left); without correction mask (middle); with correction mask (right)

The effects of the filter's sensitivity can be observed in Figures 18(c) and 18(d), when abrupt transitions from black to white or vice-versa occur. An example of signal correction using edge coefficients is shown in Figure 19.

Image pre-filtering

An example of pre-processed image is shown in Figure 20. As can be seen, the algorithm excludes most of the artifacts and substantially increases the input signal's quality. Therefore, the filtering demands for the drop detection algorithm decrease,



(a) Input image (b) Image after pre-processing

Figure 20. *Example of image pre-processing*

enabling it to be configured with light-footprint parameters. This increases both its performance and sensitivity, making it able to respond quickly to events in case of nozzle failure, with the precision of one to five droplets.

Conclusions and future work

This contribution has presented an overview of a cyber-physical system for real-time reliability testing of inkjet printheads. It is an ongoing project meant to provide high-productivity solutions for accurate examination in both graphical and functional printing applications.

The current paper has been focused on three main algorithms. Their analysis has shown promising results from both a functional and a real-time performance point of view. The nozzle auto-calibration algorithm proved to be a proper method to overcome human errors and increase the testing productivity, due to its adaptive profile based on cumulative environment information. The drop detection set of algorithms has indicated increased robustness in analyzing even the worst case scenarios, since it reconstructs obstructed information from neighboring events. Its shift of perspective from image segmentation to digital signal processing enables high-performance computing and further exploitation of the parallel paradigms. The image pre-processing algorithm has shown the potential to overcome the main drawback in the drop detection, namely the filter sensitivity, by increasing the droplet image's quality before it is transformed into a signal. All analyses have manifested positive feedback as long as there was a coherent pattern of droplets underneath the environment hazards.

The current project is part of and serves as case study for a larger-scale project – ForSyDe, a system design methodology [7]. In ForSyDe, systems are modeled and validated at a high level of abstraction, and implemented on heterogeneous hardware platforms employing massively parallel processors through state-of-the-art techniques like *design space exploration* [8], *semantic-preserving transformations* [7] or *refinement-through-replacement* [9]. The purpose of ForSyDe is to provide the designer with an intuitive framework for developing complex and *correct-by-design* systems using abstract building blocks called *processes* that communicate through *signals*.

Since this application implies analysis and manipulation of data at a very high throughput in real time, a reliable implementation is significantly difficult to achieve through conventional paradigms. For this reason, as part of future work the printhead reliability application will be fed into ForSyDe design flow and mapped to massively parallel hardware platforms (currently GPGPUs are targeted). Preliminary studies and implementations of core functionality rendered positive results [10].

Future work will include two tracks for research. The first track is related to the continuous development of the inkjet printheads reliability system by enhancing the algorithms and implementing the missing blocks in Figure 8. The second track involves implementing and optimizing an automatic design flow for modeling and implementing cyber-physical systems. The expected outcome of this project is to provide both an advanced reliability testing application, and a design methodology with a set of tools that will revolutionize system design paradigms and notably reduce a product's development cycle.

References

[1] Herman Wijshoff, The Dynamics of the Piezo Inkjet Printhead Operation. Physics reports, 491(4), 77-177. (2010).

[2] Bernd Bleymehl, A real-time Test for the Reliability of Print Heads, Dipl. Thesis, Inst. AOE, Univ. of Applied Sciences, Jülich (2010).

[3] Christian Menne et al., A Realtime Software-Based Optical Reliability Test Method for Inkjet Printheads, iPack Center Annual Workshop, School of ICT, KTH, Stockholm, (2012).

[4] Gonzalo R. Arce, Nonlinear Signal Processing: A Statistical Approach, Wiley, New Jersey, USA, (2005).

[5] Edward A. Lee, Cyber-Physical Systems: Design Challenges. Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE Intern. Symp. on. IEEE, (2008).

[6] Alan Oppenheim et al., Discrete-Time Signal Processing (Vol. 5), Upper Saddle River: Prentice Hall., (1999).

[7] Ingo Sander and Axel Jantsch, System Modeling and Transformational Design Refinement in ForSyDe [Formal System Design]. CADICS, IEEE Trans. on 23.1, pg. 17-32. (2004).

[8] Ingo Sander and Seyed Hosein Attarzadeh Niaki, Towards a Formal Software Synthesis Methodology for Embedded Multiprocessor Systems. In: Proc. of 1st Int. STEW, (2011)

[9] Seyed Hosein Attarzadeh Niaki and Ingo Sander, Semi-Formal Refinement of Heterogeneous Embedded Systems by Foreign Model Integration. Specification and Design Languages (FDL), Forum on. IEEE, pg.1-8, (2011).

[10] G. Ungureanu, Automatic Software Synthesis from High-Level ForSyDe Models Targeting Massively Parallel Processors, MSc. Thesis, School of ICT, KTH, Stockholm (2013).

Author Biography

George Ungureanu received his BSc. in Electronics Engineering and Telecommunications from "Politehnica" University of Bucharest, Romania, in 2011, and his MSc. in System-on-Chip Design at the Royal Institute of Technology, Sweden, in 2013. He is currently a PhD student in the Electronic Systems department at the Royal Institute of Technology.

Since the summer of 2012 he is doing an internship at XaarJet AB, Järfälla, Sweden. The presented project is part of both his internship and his MSc. thesis. His interests are electronic systems design and parallel platforms.