

A MATLAB-based software for realizing an affordable real-time measured and controlled ultraviolet curing system in photopolymer additive manufacturing

Xiayun Zhao^a, David W. Rosen^{a,*}

^a*The George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA, United States*

ARTICLE INFO

Article history:

Received

Received in revised form

Accepted

Keywords:

Additive manufacturing

Photopolymerization

Open source software

Real-time feedback control

Process measurement

Parallel computing

MATLAB

Cyber-physical system

ABSTRACT

As measurement science and closed-loop controls become increasingly critical in advancing additive manufacturing (AM), corresponding software development is important for physical system realization, test and implementation. This paper introduces a MATLAB-based software for real-time measurement and control of a photopolymer part's height during an in-house AM process - Exposure Controlled Projection Lithography (ECPL). A graphical user interface is created to operate the ECPL process and the self-designed interferometric curing monitoring and measurement (ICM&M) system. Parallel computing is employed to synchronize the ICM&M image data acquisition and its computational analysis. The software enables a user-friendly and cost-effective real-time photopolymer AM system, which does not require high performance computer nor prohibitive programming skills. Promoting the open source initiative, all the software codes, demo videos and example data are shared publicly in a GitHub repository. The reported software development of function modules, data structures and flow charts, could be easily extended to software programming for monitoring other AM process such as stereolithography and laser powder bed fusion processes, which may utilize some imaging approaches for monitoring the building chamber. This study also exemplifies a lab scale cyber-physical system of manufacturing process control, which is desired in the nascent and broad research area of measurement feedback control for AM.

© All rights reserved.

1. Introduction

Machine control software that inhibits real-time parameter changes using sensor-derived inputs is identified as one of the technology challenges for AM processes and equipment [1]. As the academia and industry become more active in research on measurement science and control technologies for additive manufacturing processes [1-3], the use of software for solving the related problems has significantly increased [4-9].

In academia, labs usually develop software in LabVIEW and/or MATLAB for measurement data acquisition and processing, transmitting, storing and presenting, as well as for hardware control. For example, in an open-architecture metal powder bed fusion (PBF) system equipped with in-situ process measurements, a LabVIEW-based software code takes as input the PBF process requirements (i.e., layer number and thickness) and dictates all aspects of depositing each powder layer by controlling the movements of the platform, the powder and the laser [7]. The National Institute of Standards and Technology (NIST) has also developed an open architecture AM research platform, which implements software interfaces and data acquisition for observation and analysis of melting and solidification of metal powders with process metrology tools [10-11]. NIST anticipates that the AM community will benefit from such a test platform to implement, test

and validate a real-time and closed-loop control of metal-based AM processes. This paper could pave the way for a similar testbed for polymer-based AM processes.

In industry, Materialise, one software backbone company in the additive manufacturing community, develops an inspector software, which is new to the market and offers a control tool that allows users to analyze data during all stages of the metal 3D printing production process for meeting predetermined quality standards [12]. Inspector optimizes image processing for efficiency in post-build analysis and is capable of processing more than 4000 images in minutes. The software represents the full digital thread, giving metal machine manufacturers the ability to develop, implement and manage each step of the 3D printing process.

This paper reports a MATLAB-based software for real-time measurement and control of a photopolymer part's height during an in-house AM process - Exposure Controlled Projection Lithography (ECPL), which is introduced in Section 2. Then, the design and development of the software is presented. Section 5 describes some examples of offline implementation, and an example of real-time ECPL process measurement and control enabled by parallel computing.

* Corresponding author. Tel.: 404.894.9668; fax: 404.894.9342.

E-mail address: david.rosen@me.gatech.edu (D. Rosen), xiayun.zhao@gatech.edu (X. Zhao).

2. Overview of the system and algorithms

As shown in Fig. 1, the physical system consists of an photopolymerization based lithographic additive manufacturing system – ECPL [13], and an interferometric curing monitoring and measuring system (ICM&M) [14].

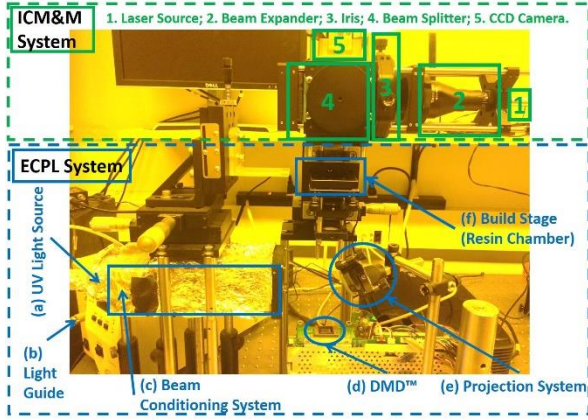


Fig. 1. System overview: ECPL process and ICM&M system.

While implementing the ICM&M method for the ECPL process measurement, a video of interferograms is used to obtain imaging and optics data which provide grayscale signal for analysis determine the cured height. The sensor model and algorithms are detailed in the authors' papers [14-15].

To use such information for ECPL process control, images from the AM processes must be processed at a speed sufficient to capture the process dynamics. The large amount of data being generated and made available by the ICM&M method for the ECPL process continue to grow with the measurement region and part size, driving the need for efficient data analysis and high performance computing tools to enable appropriate measuring.

Please note that a regular computer processing units (CPUs) was used in this study, although faster systems such as graphical process units (GPU) could be more powerful.

3. Design a software for the integrated system of ECPL and ICM&M

For solving problems in metrology, software has been increasingly used for data acquisition and processing, transmitting, storing and presenting measurement results, along with auxiliary infrastructural device and information [16]. Requirements for measurement software include compatibility with hardware, adequacy of computation resource, integrity of code and configuration parameters. In real time scale, the permissible latency time for getting a measurement result is another important consideration. Errors of the software itself should be insignificant as compared to the transformed errors of input data and to the methodical errors of the algorithm.

For ECPL process measurement and control, a software is needed to interface the ECPL system's hardware equipment including the UV lamp, DMD and CCD camera for process operation, measurement and control. In Fig. 2, the brown dashed lines indicate the communication between the software and hardware. The double arrows between the app and the camera mean that the software could both send instructional signals to the camera and receive image data from the it. The one-way arrow to UV light source and to DMD, respectively, means that it only writes into the lamp and DMD without data

feedback. As a console for transmitting commands and data, the software is also intended for all numerical computing needed for the process control.

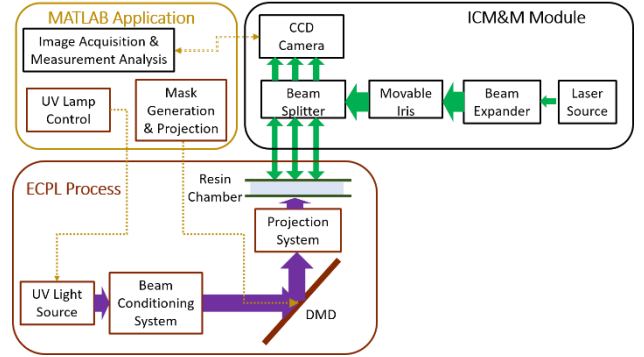


Fig. 2. The integrated ECPL system: ECPL process, ICM&M module, and MATLAB software for ECPL M&C

The MATLAB application is referred to as “ECPL-M&C software” in the study, and designed with the specific functions as below.

1. Provides a human machine interface (HMI) to operate the system
 - (1) For the ECPL process
 - a) Turns on / off the ECPL UV lamp (OmniCure® S2000) to start / stop the curing process.
 - b) Sets the UV iris level to manipulate the irradiation intensity for ECPL curing.
 - c) Generates cross-section bitmaps for the target 3D object.
 - d) Projects the time sequence of bitmaps onto DMD to obtain the specified thickness for each part cross-section.
 - (2) For the ICM&M module
 - a) Starts / Stops the ICM&M camera.
 - b) Selects a region of interest (ROI) to analyze and measure.
 - c) Starts / stops image acquisition and analysis.
 - d) Writes the acquired images into a video file for recording the ECPL process dynamics.
 - e) Reports and saves process data and measurement results.
2. Performs all the computation for the ECPL process measurement and control.
 - (1) For measurement
 - a) Implement the ICM&M sensor model and algorithms to measure the ROI pixels cured height
 - b) Complement the ICM&M algorithms with data analysis of the estimated parameters and the resultant measurements to evaluate the cured height more accurately.
 - (2) For control
 - a) Implement the time control algorithm to derive the control input of exposure time for each bitmap.
 - b) Implement the intensity control algorithm to derive the control input of exposure intensity (i.e., UV iris level) for each bitmap. This task is out of the work scope of the research and will be incorporated in the future work.

4. Development of the ECPL-M&C software

4.1. Software tasks

A graphical user interface using the graphical user interface development environment (GUIDE) of MATLAB was created to implement the ICM&M method for the ECPL process. The application was designed to perform the following tasks.

1. Visualization

Build a graphical user interface to visualize the process interferograms and online measurement results.

Provide user control components such as pushbuttons and edit boxes to operate the ECPL process and to streamline the process with the ICM&M acquisition and measurement analysis.

2. Hardware connectivity

Connect and communicate between the software application and hardware equipment including the ultraviolet (UV) lamp and DMD in the ECPL system as well as the CCD camera in the ICM&M system.

3. Data management

Create a MATLAB memory map file to log the acquired interferograms data and timestamps, process parameters such as the UV lamp iris level and exposure duration, and measurement parameters such as measurement period and calibrated refractive indices.

4. Numerical computation

Implement the ICM&M algorithms, and perform data analysis required for best estimation of the cured height profile of the ECPL cured part.

5. Report generation

Document and save into MAT-files all the data, curve fitting estimated parameters for the sensor model, along with the measurement results of the cured heights. Display and save the results of time-height curve and ROI height profile in figures.

4.2. Modularity and scalability

As shown in Fig. 3, the application can be functionally modularized into two main parts – process measurement with ICM&M (green parts in the figure) and process control of ECPL (blue parts in the figure). The ICM&M part is further divided, based on usage, into offline ICM&M (brown in the figure) and real-time ICM&M (purple in the figure), both of which employs the same algorithms to analyze the interferograms data extracted from offline replay and real-time acquisition, respectively.

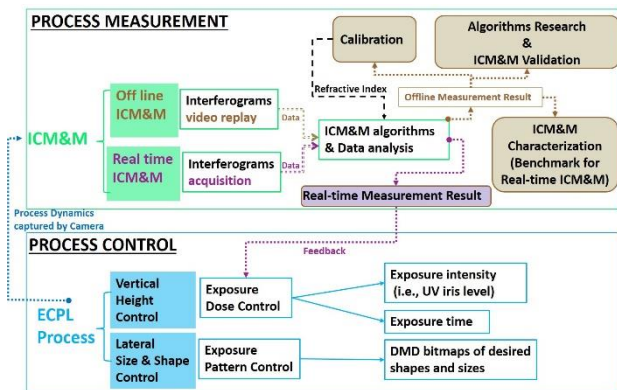


Fig. 3. MATLAB application for the ECPL process measurement & control: functional modules

The ECPL process part can be broken down into lateral control and vertical control sections. Each module is designed with considerable scalability to extend the software capability in future for full-field

measurement and comprehensive control in fabricating complex 3D objects for real-world applications. Please note that the current program is created as even-driven with functions being called upon as needed. This allows additional parts to be added to the program without affecting the functionality of the original program.

4.2.1. Software module for the ICM&M

The offline ICM&M utilizes a video record of interferograms captured when the ECPL process was curing a part to measure that cured part. The three brown blocks on the upper-right corner of Fig. 3 states that the three main uses of the offline measurement result: 1) to calibrate the refractive index as explained in the ICM&M sensor model and calibration process; 2) to serve as a simulation analysis tool for the ICM&M algorithms development as shown in the “Algorithms research & test”; 3) to provide a benchmark unveiling the full measurement capability and accuracy without computation power limits effects that would make the real-time ICM&M underperform.

The real-time ICM&M features in-situ interferograms acquisition. In MATLAB, the “Image Acquisition Tool” provides an efficient workflow to solve the challenging imaging acquisition problem using the ICM&M system’s GigE vision camera. To use the specific ICM&M camera in MATLAB, the right hardware support packages for both “IP Camera” and “Gige Camera” must be installed so that the software can recognize the camera correctly and access to its attributes settings properly. It is worthy to point out that we initially used the “OS Generic Camera” hardware package which assumes wrongly the camera as “winvideo” and could not access fully or correctly to its attributes; and consequently we couldn’t capture the whole curing area with a reasonable resolution. Another issue with the camera misidentification was that it was not stable in the camera output formats which cause trouble in our experiment of ICM&M. Appropriate handling of the ICM&M camera as a Gige IP camera helps integrate the ICM&M hardware with the ECPL system neatly in the software app. The application was ultimately programmed to be able to configure the camera attributes and acquire images with triggers. For the image acquisition, a timer function is used to regulate the acquisition speed in MATLAB.

The “ICM&M algorithms & Data analysis” section is coded to execute the ICM&M algorithms of moving horizon exponentially weighted Fourier curve fitting and numerical integration, along with some necessary data processing and analysis to better estimate the cured height. It can analyze the intensity data for all pixels within the preset region of interest (ROI), which could be selected interactively by mouse dragging a rectangle on the interferogram or specified programmable by defining a matrix of interesting pixels’ coordinates. For instances, after an interactive placement of a rectangle on the current axes, the analysis module can automate the data processing and measurement analysis for the rectangle’s center pixel or corner pixels or horizontal centerline or vertical centerline or all of them at the same time. Alternatively, if an area of pixels is specified as the ROI in the code, the analysis module can output the area height profile correspondingly. It is worth to note that specifying ROI pixels in the code before-hand instead of in the GUI platform in real-time can facilitate continuous acquisition and logging. Conclusively, the application features wide measurement types including single point, discrete multi-point, line profile and area profile vertical height. It is scalable to full-field measurement as long as the computer has sufficiently fast processor and large memory.

4.2.2. Software module for the ECPL: principle and practice

As stated, the ECPL process control consists of two general dimensions' control: the lateral control and the vertical control. This research focuses mainly on the vertical control and would address the lateral control only to some extent as necessary, because the vertical control is not as straightforward as the lateral control and presents more research interest and challenges. Besides, in cases of curing a 3D parts which can be discretized into vertical voxels, controlling the vertical height profile would have virtually rendered the lateral shape and surface profile.

The lateral control module aims to manipulate the size and shape of bitmap displayed on DMD which determines the cross-section of the cured part. It can provide a sequence of bitmaps with different sizes and/or shapes required to form a complex 3D object. In this study, experiment of curing rectangle blocks is used to test and validate the process, but please be noted that a wide range of geometrical shapes bitmaps (e.g., circles) could be generated and employed in real applications and the software can easily be adapted and extended for more complex 3D objects curing.

The vertical control module, which is the primary research objective, controls the cured height of a voxel. A voxel is defined as a block growing on the resin substrate which is divided into pixels that

are mapped to the interferogram pixels, so that the lateral size of the cured part corresponding to one interferogram pixel determines the lateral area resolution for a voxel.

In the study, a lateral control session with rectangle bitmap specification and generation, and a simple on-off control with the real-time ICM&M feedback intended for the vertical height control are included in the software application.

4.3. Graphical user interface

Fig. 4 shows the application's graphical user interface (GUI). To start with the application, one should select from the left-bottom dropdown menu "Real-time Measurement" or "Offline Measurement", for the former involves with real-time interferograms acquisition and simultaneous online measurement while the later replays off line an already acquired video of interferograms and performs ex-situ measurement.

"Measurement Parameters" as shown in the upper-right panel are always required beforehand and one can use the "Default" parameters which may provide a good set of starting values but may still need modifications, especially for the refractive indices values which are subject to change with materials and disturbances (e.g. oxygen diffusion and inhibition).

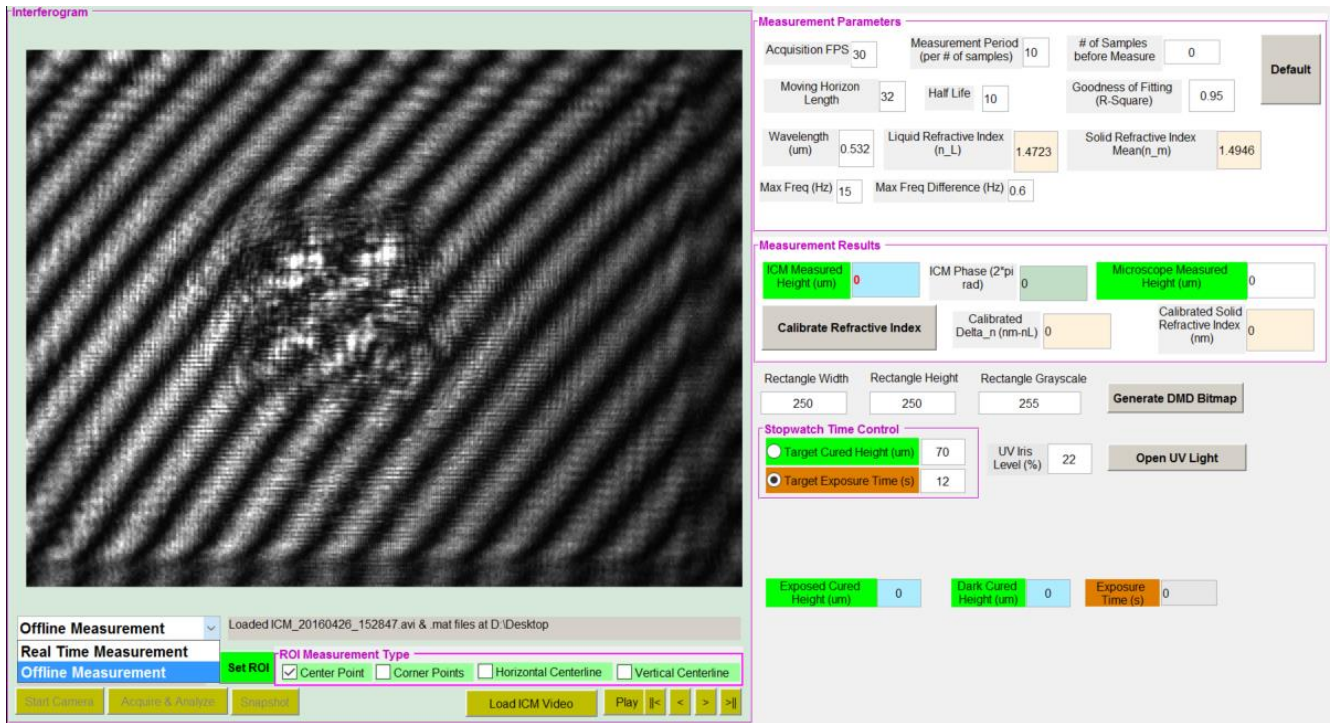


Fig. 4. MATLAB application for the ECPL process measurement & control: Graphical User Interface

In the "Real-time Measurement" practice, one firstly presses the button "Start Camera" to set up the video input object in MATLAB and to configure the ICM&M camera setting, and can immediately preview images in the GUI axes. Secondly, click "Set ROI" to notify the application the pixels of interest for measurement analysis. Herein the ICM&M part is ready and the ECPL process part should be started with inputting the bitmap size and grayscales into the "Rectangle Width" and "Rectangle Height" boxes followed by pushing buttons of "Generate DMD Bitmap". Please note that the application provides a rectangle bitmap of user-specified width, height and grayscale, for the sake of experiment simplification and validation efficiency, and can be easily extended to a series of various shape bitmaps in future. Two

methods of exposure time control are provided for deciding when to turn off the UV lamp. One is simply a stopwatch control with preset target exposure time, the other a basic negative feedback control - "On/Off" control decided by whether or not the measured cured height has reached the setpoint of cured height. The user must select one of them in the experiment. Also, the UV iris level should be specified, otherwise "22" is used by default. Thirdly, press "Acquire & Analyze" to get ready the image acquisition and measurement computation. Lastly, user needs to click "Open UV" to turn on the UV lamp, the UV light beam will be shaped by the DMD displaying the previously generated rectangle bitmap, and a rectangle block will be cured in the

resin chamber. Meanwhile, the entire ECPL process should have been captured by the ICM&M module.

If the ROI is empty, only real-time acquisition without online measurement will be performed. In the validation of ICM&M method, due to the computer limitation, we did not set ROI for online ICM&M which would cause severe loss of frames during image acquisition and impair the performance evaluation of the ICM&M method. However, we later improved the software code with parallel computing and presented the real-time ICM&M results in the thesis.

In case of “Offline Measurement”, instead of starting camera, one firstly should load an acquired video of interferograms and associated timestamp file by pressing the button “Load ICM Video”. A toolbar of replaying video is provided for user to play or stop, and to play the first frame, the previous frame, next frame and last frame as shown at the bottom of GUI. Click “Play” and stop at the desired starting frame for measurement analysis, click “Set ROI” and “Play” again. Then the application will perform measurement analysis while playing the frames, which could be taken as a simulation of the “Real-time ICM&M” doing acquisition and analysis simultaneously.

4.4. Data structure

In MATLAB, “struct” and “cell” arrays are the most commonly used containers for storing heterogeneous data of different types and sizes. One most important data structure – the resultant data structure in the software program is introduced as below.

The measurement result, as stored in the variable “MeasureRet”, returns all the rolling fit coefficients, processed coefficients and online estimated heights. It is an nPOI-by-1 structure array of all points measurement. Specifically, the result variable “MeasureRet” is a structure array including the following data:

- 1) nPOI: number of points of interest
- 2) RunNo: number of runs of online parameter estimation
- 3) “PixelHeightWidth”: pixel’s coordinate vector [height; width]
- 4) “PixelGrayscaleData”: point data, i.e., time-series of pixel grayscale
- “PixelGrayscaleData_Smooth”: smooth data using a moving average filter.
- 5) “FittedCoeffs”: coefficients returned by each run of rolling fit for each POI
- “fourier1” returns 4 coefficients $y = a_0 + a_1 \cos(px) + b_1 \sin(px)$
- Note: first set of coefficients is the initial values
- The length of coefficients = $4 * (\text{RunNo} + 1)$ due to the initial value
- FittedCoeffs = [a0; a1; b1; p].
- 6) “ProcessedCoeffs”: Processed coefficients, will grow to be a 3-by-RunNo matrix

“ProcessedCoeffs” = [I0; I1; p], where I0 is the estimated baseline amplitude (DC), I1 is the estimated fringe amplitude (AC), and p is the estimated frequency.

Note: $y = I_0 + I_1 \cos(px + \phi)$, the phase angle ϕ is not of interest here, hence not stored in the data.

- 7) “CureFlags”: Flag the curing window, i.e., mark the beginning of curing

- 8) “Times”: RunNo-by-1 matrix, array of run time for each POI
- 9) “Heights”: RunNo-by-1 matrix, estimated height at each run for each POI

- 10) “Freq_w”: RunNo-by-1 matrix, the real frequency “ ω ” of $I_m = I_0 + I_1 \cos(\omega t + \phi)$ at each run for each POI

4.5. Parallel computing

An easy and efficient data-parallel programming is essential and desired to fulfill the performance of real-time ICM&M and ECPL process control. MATLAB® provides two main ways to take

advantage of multicore and multiprocessor computers. By using the full computational power of your machine, one can run the MATLAB applications faster and more efficiently. One can run multiple MATLAB workers (MATLAB computational engines) on a single machine to execute applications in parallel, with “Parallel Computing Toolbox™” [17]. This approach allows one more control over the parallelism than with built-in multithreading, and is often used for coarser grained problems such as running parameter sweeps in parallel.

4.5.1. Some key variables

4.5.1.1. “status”

“status” is a global variable which conveys three important statuses – “measurement status”, “lastWriteFrameIdx” and “workerReady”.

- (1) “measurement status”. 1: running, 0 stopped.

- i. Used in the timer function “processMeasureTimer”: if measurement is stopped, stop acquisition and writing frames and just preview.
- ii. Used in “icm_main_worker.m” while loop, if measurement is running, keep the loop of measurement analysis.
- iii. In “pb_AcquireAVI_Callback.m”, set to be “0” when push button “Stop Acquisition”
- iv. In “pb_AcquireAVI_Callback.m”, set to be “1” when push button “Acquire & Analyze” and calls the function “icm_init_mem_file.m” which actually initialize the status to be “1”

- (2) “lastWriteFrameIdx”:

Used to mark the index number of previous frame for continue numbering the subsequently acquired frame.

- (3) “workerReady”

In “pb_AcquireAVI_Callback.m”, set to be “0” when push button “Acquire & Analyze” and calls the function “icm_init_mem_file.m” which actually initialize the status to be “0”.

Then still in “pb_AcquireAVI_Callback.m”, it creates the batch job “icm_main_worker.m” which gets started and when ready tells GUI that worker is ready by setting the status to be “1”.

In “pb_AcquireAVI_Callback.m”, it waits until the status is set to be “1” and continues to proceed to notify the timer function for image acquisition by initializing the frame index, uvStatus and acquiring status, and image data matrix, and by enabling the UV Lamp Switch Button. Only when UV lamp is turned on, the acquisition would really start in the video timer function so that it can acquire the entire process from UV curing start till after curing stop.

4.5.1.2. “handles.mmf”

Memory-mapping is a mechanism that maps a portion of a file, or an entire file, on disk to a range of memory addresses within the MATLAB® address space. Then, MATLAB can access files on disk in the same way it accesses dynamic memory, accelerating file reading and writing. Memory-mapping allows you to work with data in a file as if it were a MATLAB array. In the real-time ICM&M app, the memory map file “icm_comm.dat” plays a role of cache. The cache file is used for temporary storage of data and can be accessed more quickly than the main memory.

The global accessible variable “handles.mmf” stores the acquired images data and can be retrieved immediately by parallel thread for measurement analysis. It is initialized in the pushbutton “Acquire & Analyze” callback function, and written in the video timer function, and read in the batch job for measurement analysis.

4.5.2. Flow chart

Below presents a flowchart of the software to achieve real-time acquisition and measurement analysis of the interferograms from the ICM&M system for the ECPL process.

1. Before acquiring images using the Image Acquisition Toolbox, create a video input object.
2. Firstly, start camera and initialize acquisition parameters, region of interest (ROI) to measure, UV Lamp Status.
3. To generate timer events, we specify two things: what happens when it occurs, and how often it should occur. The `TimerFcn` property specifies the callback function to execute when a timer event occurs. A timer event occurs when the time period specified by the `TimerPeriod` property expires.
The callback function is responsible for triggering the acquisition and storing the frame into the AVI file. More details on how to use this callback can be found in the documentation.

4. The Timer Callback Function

The following is a description of the callback function executed for each timer event.

- (1) Snapshot a single frame
 - (2) Determine whether to keep the frame for analysis. If user has not pushed the button to start analyze and to open UV light yet, just preview without saving the frame for measurement analysis.
 - (3) In the timer function, when GUI pushbutton of "Acquire and Analyze" is pressed, the acquisition is officially started, and the GUI data "handles.dp.acquiring" is set true notifying the timer function to start acquisition rather than preview only. The timer function acquires latest frame by snapshot and display it on GUI axes, meanwhile it writes the memory map file "handles.mmf" which data include image frame data, frame index, frame time stamp (time elapsed from the 1st frame), UV iris level (a metric for UV light intensity), UV lamp status of "On" or "Off", "snapTic" which starts a stopwatch timer to measure the delay between writing and reading the frame.
5. The pushbutton "Set ROI" will allow user to specify the region of interest (ROI) which include all the pixels to be measured for height profile. The ROI is a matrix of pixels coordinates [Height; Width]. The app will extract the time series of grayscales of the ROI pixels and measured the corresponding voxels height.
 6. The pushbutton "Acquire & Analyze" will perform the following tasks.
 - (1) Initialize memory map file for GUI: "handles.mmf"
 - (2) Start a batch processing job as a main worker for data analysis and measurement calculation : "handles.job", which runs MATLAB script "icm_main_worker.m" with the curve fitting function "icmFit2.m" on the main worker.
 - (3) To expedite the measurement related analysis and computation, more workers may be needed to make into a parallel pool for the job in addition to the main worker running the batch job itself. The script "icm_main_worker.m" uses this pool for execution of statements "parfor" that is inside the batch code. The "parfor" loop iterates online parameter estimation for each pixel calling the core function of curve fitting "icmFit2.m" which is the most time-consuming.

Because the pool requires N (e.g., $N=2$ in this study) workers in addition to the worker running the batch, there must be at least $N+1$ workers available on the cluster. One does not need a parallel pool already running to execute batch; and the new pool that batch creates is not related to a pool one might

already have open. The default value is 0, which causes the script or function to run on only a single worker without a parallel pool [18].

Therefore, a number of workers are assigned in the batch job as fitters to do the online parameter estimation by curve fitting for pixels in parallel loops. The fitters number is recommended to be the number of computer cores in local cluster to optimize the computer resource without causing adverse effect of overhead cost such as communication time. In this study, two fitters are assigned for "parfor" loop in MATLAB to calculate two pixels' cured height simultaneously. If multi-core computer with more computation power is provided, more workers might be available to accelerate the computation for more pixels in ICM&M analysis.

- (4) Initialize the acquired frame index, UV status to be "0" which means the UV is not on yet, and acquiring status to "1" which will notify the video timer function to start acquisition.

The main worker function measures the height of each ROI pixel in a timely fashion per measurement period (e.g., 10 frames per run of measurement) and reports the time delay from writing to reading each frame, i.e., from acquiring to analyzing each frame, so as to provide insight about how the measurement is keeping in pace with the data acquisition. It also saves the metadata of ICM&M constant parameters, dynamic data including all frame data - time sequence of grayscales and frame index, measurement results including all curve fitting parameters and results, and matrix of time delays.

7. The pushbutton "Stop Acquisition" will perform the following tasks.

- (1) If the real-time measurement is running, notify the video timer function to stop acquisition and meanwhile notify the main worker to stop measurement
- (2) Stop the camera and write the acquired images into a video file
- (3) Save the constant parameters
- (4) Wait for the batch job to be completed and display the result diary
- (5) Denounce the mask of selected region of interest for measurement

8. When user stops the camera, all the frames are written into a video file by the `writeVideo` function.

The flowchart diagram is illustrated in Fig. 5. The blue shape represents MATLAB graphical user interface (GUI) controls such as push buttons, edit boxes, and plot axes. The blue arrows depict the typical user manual operation that implements the ICM&M for an ECPL curing process which is shown in the brown block. The purple arrows direct toward the associated key functions. There are two main computing threads (shaded green boxes in the figure) running in the MATLAB: one is the video timer function for image acquisition at specified frequency, the other the batch process job which off-loads the execution of long-running computations of scripts and functions for online measurement analysis and controller computation in the background. The red fonts and arrows denote the key data of a memory map file that enables the communication between the parallel computing blocks. As shown in the brown shaded box related directly to the ECPL process, the user turns on the UV lamp by pushbutton "Open UV", the ECPL process gets started, the image acquisition is also triggered, the memory map file is written and immediately read by the batch job for measurement. The online measurement is feedback into the controller which would calculate the input for the actuation.

After the UV lamp is closed, dark curing occurs in place of exposed curing in the ECPL process, and the user could wait for a few seconds

before stopping the acquisition to capture the entire process including the dark curing.

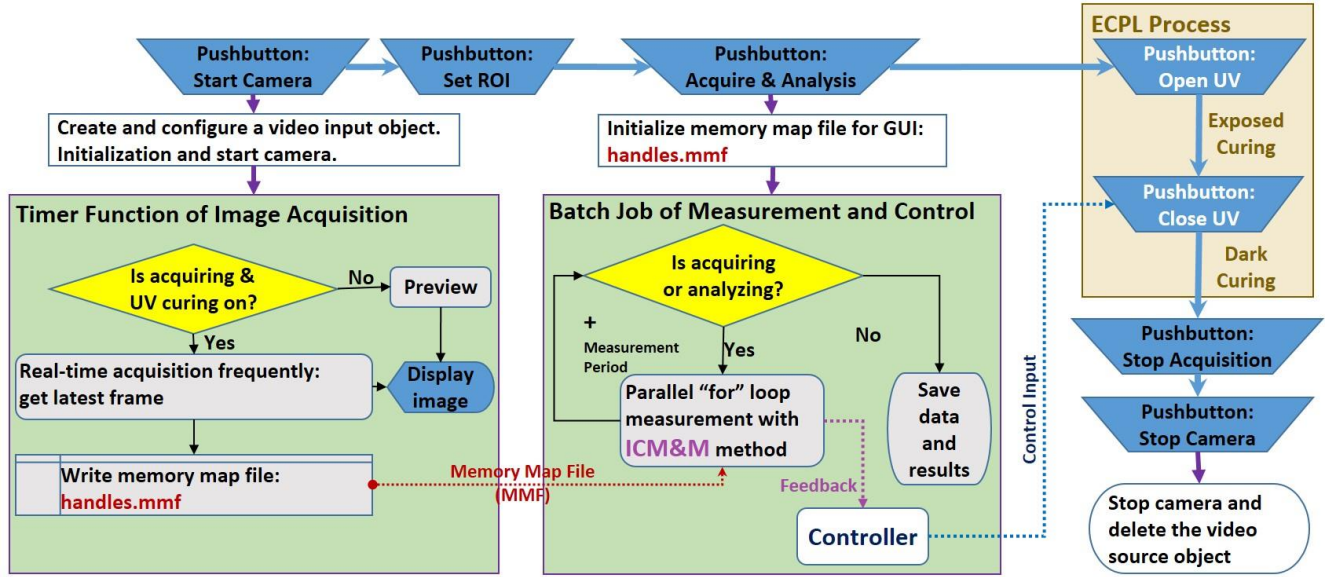


Fig. 5. Flow Chart: Parallel Computing of Real-time measurement and control for the ECPL process

4.5.3. Key functions

This section lists all the key functions in the MATLAB based ECPL-M&C software.

4.5.3.1. Function: “icmFit2.m”

The function is for adaptive curve fitting. Detailed algorithms and parameters are studied in the authors’ previous publication [15].

4.5.3.2. Function: “Real_Time_ICM_processMeasureTimer.m”

The function is used for real time ICM&M data acquisition (i.e., interferogram acquisition) and logging. It also tracks the process status including the UV lamp status.

4.5.3.3. Function: “icm_main_worker.m”

The function is used for real time ICM&M data analysis, measurement and control computation.

4.5.3.4. Function: “icm_init_measure_ret.m”

The function is to initialize the structure array of measurement result for the chosen POI.

4.5.3.5. Function: “icm_init_mem_file.m”

The function is to initialize the memory map file for logging the real-time data which can be fetched by the parallel thread of measurement analysis and controller computation.

4.5.3.6. Function: “icm_init_fit_ret.m”

The function is to initialize the structure array of curve fitting result for one measurement cycle.

4.5.3.7. Function: “icm_set_uv_status.m”

The function is to display UV pattern on DMD, and to control UV lamp “On/Off”.

4.6. Computation environment

The application is executable in MATLAB R2015b for 64-bit Operating System and can be used in both experiment and post analysis. The real-time acquisition of the ECPL process was done in-situ on the lab desktop computer with a processor of Intel® Core(7M) i7 CPU 870 @2.93GHz 2.94GHz and an installed memory (RAM) of 16.0 GB (8.00 GB usable). All the offline ICM&M analysis was done on an ex-situ Lenovo laptop with Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz 2.6 GHz and an installed memory (RAM) of 8.00 GB. Provided a more powerful multi-core processor and a high-speed camera, the ICM&M is expected to be able to run faster and more accurate measurement online with a full-field measurement capability if necessary.

5. Examples of the software implementation

Two examples are provided in this session to instruct users how to use the software for, respectively, (1) offline measurement of a cured part [19] with an existing ICM&M video file and (2) real-time measurement feedback control of a cured part during the photopolymerization process given a physical system available [20].

5.1. Offline implementation

Two demo videos are provided in the GitHub repository (<https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Offline%20Implementation/Demo>), showing how to use the software to analyze off line the interferometric video and associated data for measuring the photocuring process in terms of cured thickness. Please note that two different samples are used in the videos, respectively. A smaller curing part experiment is used to demonstrate the software’s capability of measuring multiple pixels in a full field.

Besides, an example with meta data and analysis results is provided for users to test or mimic the analysis (<https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for->

[photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Offline%20Implementation/Data](#)). Please note that the analyzed video was acquired in a real-time experiment of the ECPL process. To implement this example, a physical system of the same setup as in the authors' lab is NOT required. One could just import the video and "mat" file in the folder (<https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Offline%20Implementation/Data/LabData>) and specify the ROI as per the "ROI_FF.png" in the MATLAB code "pb_SetROI_Callback.m", then run the software to measure the ROI's pixel(s)' cured heights. If interested, one can refer to the results in the folder "ResultsData" (<https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Offline%20Implementation/Data/ResultsData>). For detailed interpretation of the results, please refer to the author's dissertation [21], where this example is referred as "Validation Group #2 Experiment #4" in Chapter 8 and Appendix. The corresponding results are also reported in a journal paper [19].

5.2. Real-time implementation

This session presents example data, experiment demo and instructions for performing a real-time measurement and control experiment with the GUI software developed as above. Please note that the corresponding experiment design and results are detailed in the authors' paper [20].

5.2.1. Example data and analysis in the real-time implementation

To showcase the real-time experiment data and analysis, a real-time acquired video and real-time logged metadata are provided in the author's GitHub (link: <https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Offline%20Implementation/Data/LabData>).

[manufacturing/tree/master/Examples-Metadata/Real-time Implementation/Data](#)).

In this real-time experiment, only one pixel (Height: 225; Width: 285) in the sample was monitored online due to limited computation resource. The real-time monitored time-sequence of the pixel's grayscale and measurement result are shown in Fig. 6.

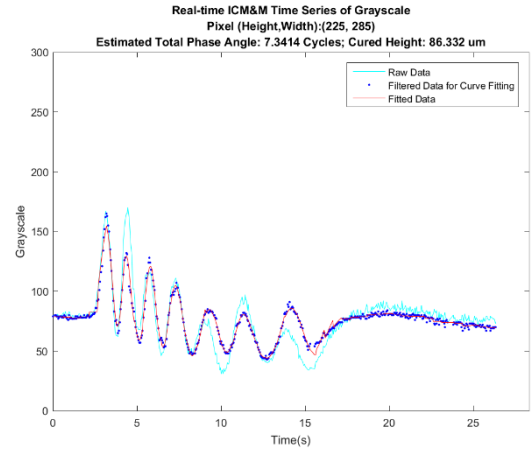


Fig. 6. Online measured pixels' data dynamics and measurement results in the real-time ICM&M analysis

The increasingly large quantity of data that must be handled creates computational challenges and potentially leads to longer run times. Computational speed and measurement run time greatly affects the usability of ICM&M developed for ECPL process. In this example, the real-time computer and software performance is reflected in the latency plot as shown in Fig. 7. The latency induces delays in measurement and control as illustrated in Fig. 8. For error analysis and more detailed interpretation of the results, please refer to the author's dissertation [21], where this example is referred as "Group 1 Sample 06" in Chapter 9 and Appendix B.

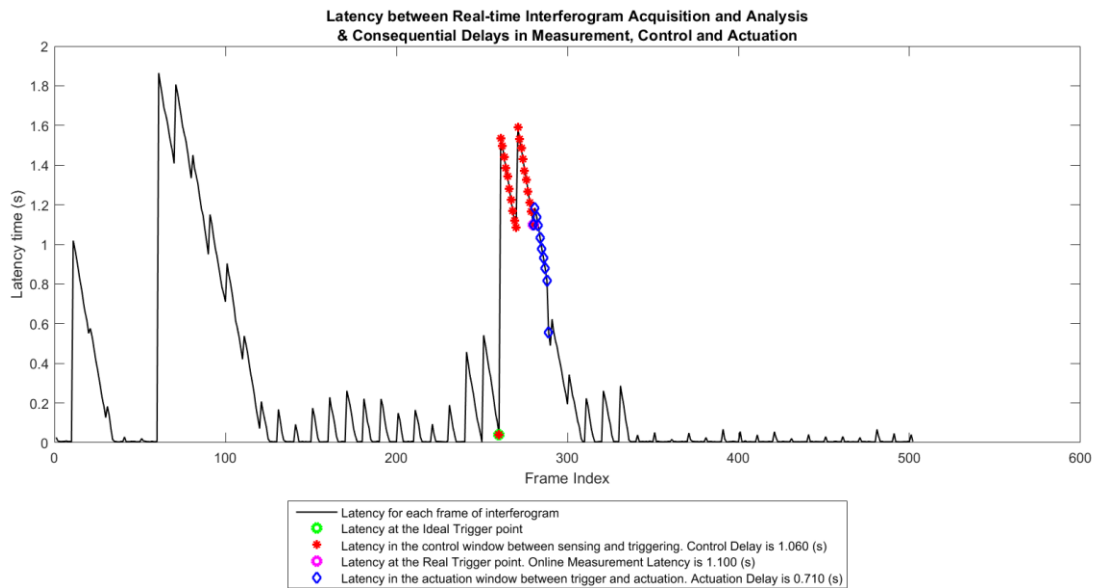


Fig. 7. Latency between real-time interferogram acquisition and analysis, and its indication for computation fluctuations that could account for delays in measurement, control and actuation

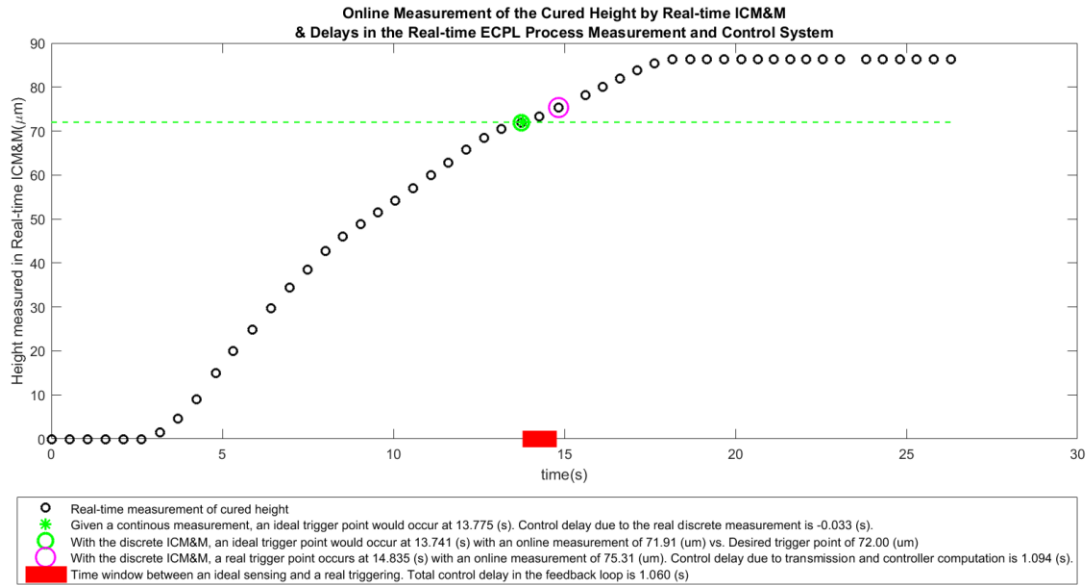


Fig. 8. Real-time ICM&M measurement and analysis for online delays in the ECPL process measurement and control system

Furthermore, to implement this example for offline analysis, one could import the video and real-time metadata files, and play the video to simulate a real-time photocuring process measurement. Then, one could specify the ROI as per the “ROI_FF.png” in the MATLAB code “pb_SetROI_Callback.m”, then run the software to measure the ROI’s pixel(s) cured heights. If interested, one can refer to the results in the folder “ResultsData” (https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Real-time_Implementation/Data/ResultsData).

5.2.2. Real-time experiment demo and instructions

To implement a real-time experiment with the ECPL machine and ICM&M system, a physical system of the same setup as in the authors’ lab [13] is required. A demo video is available in the GitHub repository (https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing/tree/master/Examples-Metadata/Real-time_Implementation/Demo) for showing the real-time lab experiment. One could also adapt the codes and parameters accordingly to the user’s specific photopolymerization based AM machine.

Experiment instructions are given as below for users who are interested in replicating the experiments or the system so that they could generalize, extend and utilize the concepts developed in the authors’ research for real-time photopolymer additive manufacturing process measurement and control.

Experiment Group # 1: 1 pixel, 22% UV [20]

Sample 1: 250×250 square

Instructions:

1. Connect UV lamp, camera, and DMD
 2. Open MATLAB, run “Real_Time_ICM”, measure & control online by following the steps as below.
 - (1) Set the measurement period box value “10” (default already), UV light “22”
 - (2) The online acquired interferograms are simultaneously written into the global variable “g_all_frame”, which is initialized in the matlab code “pb_AcquireAVI_Callback.m”, and user could set the number of maximum frames to be acquired in that code as shown in Fig. 9.
- For UV=22% and curing 80 μm , 500 frames will do. Hence, one could change the frame number at Line 72 (highlighted in blue) to be “500”.
- (3) Input “250” in the boxes of “Rectangle Width” and “Rectangle Height”, and “Generate DMD bitmap”
 - (4) Make sure to CHECK “Target Cured Height” and UNCHECK “Target Exposure Time”, this is already by default but better make sure. Set the Target Cured Height as “80”.

```

pb_AcquireAVI_Callback.m  X +
51  %% Start the icm main worker
52  disp('Starting ICM main worker')
53  set(handles.st_InterferogramStatusBar,'String','Starting ICM main worker');
54  pause(0.5);
55  g_job = batch(@icm_main_worker, 1, (handles.cp),...
56  'Pool', handles.cp.numFitter,...
57  'AttachedFiles', {'icmFit2.m'})...
58  );
59  wait(g_job, 'running');
60  disp('ICM main worker started, wait until worker is ready')
61  set(handles.st_InterferogramStatusBar,'String','ICM main worker started, wait until worker is ready');
62  while g_mmf.Data(1).status(3) == 0
63      pause(1)
64  end
65  disp('icm main worker is ready!')
66  set(handles.st_InterferogramStatusBar,'String','ICM main worker is ready! Click "Open UV" to start ECPL and ICM&M.');
```

```

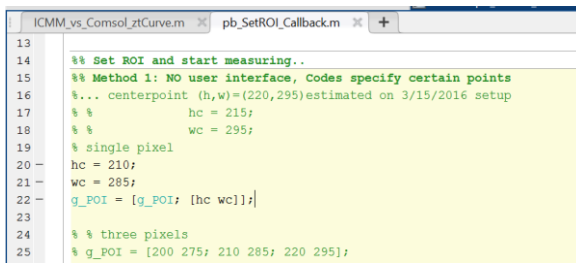
67
68
69  g_frameIdx = 0;
70  g_acquiring = 1;
71  g_uvStatus = 0;
72  g_all_frame = zeros(handles.cp.imH, handles.cp.imW, 1, 1000); % HOME computer & camera
73
74  set(handles.pb_OpenCloseUV,'Enable','on');
```

Fig. 9. Set the number of maximum frames to be acquired

(5) "Start Camera"

(6) "Set ROI" (1 pixel: H=210, w=285)

Step: In "pb_SetROI_Callback.m", change the g_POI as shown in Fig. 10.



```
13
14 %% Set ROI and start measuring..
15 %% Method 1: NO user interface, Codes specify certain points
16 %... centerpoint (h,w)=(220,295)estimated on 3/15/2016 setup
17 % %          hc = 215;
18 % %          wc = 295;
19 % single pixel
20 hc = 210;
21 wc = 285;
22 g_POI = [g_POI; [hc wc]];
23
24 % % three pixels
25 g_POI = [200 275; 210 285; 220 295];
```

Fig. 10. Set region of interest (ROI) for the measurement

7) "Start Acquisition" and wait until the pushbutton "Open UV Light" is enabled. This may take one minute or longer.

(8) "Open UV Light"

(9) After UV light is off, wait for 3 seconds, click "Stop Acquisition", and hold on till the status bar under the image shows "Main worker Returned measurement and analysis results."

Note: Do NOT click "Stop Acquisition" again if it still continues acquiring because the acquisition will stop automatically when 500 frames are acquired.

(10) Close the app.

6. Conclusion

This paper documents the development of a comprehensive MATLAB-based software platform for measurement and control of the ECPL process. To implement ICM&M method, the application program was designed and created in MATLAB Graphical User Interface Development Environment (GUIDE). The app can be deployed onto the physical system integrating the ECPL and ICM&M automating the ECPL process by controlling the ultraviolet lamp and DMD mask display as well as synchronizing the ICM&M which acquires and analyzes interferograms online. The parallel computing toolbox was employed in MATLAB to accelerate the computation. Importantly, the display of interferogram frames is temporally precise and is achieved without a substantial sacrifice in temporal performance of other key functions such as data acquisition and ICM&M analysis.

Cyber-physical systems are fundamentally defined as integrations of computation and physical processes [22], and can range from minuscule to large scale. To address the demanding need for real-time process control in AM industry, this study demonstrates a lab scale cyber-physical system, which efficiently connects the photopolymer-based AM system components with a cost-effective MATLAB based software. The ECPL process measurement and control benefits from the developed MATLAB application, a comprehensive and automatic platform for effortless operation of the AM process and online monitoring and measurement in just a few clicks. It is hoped that this software will improve productivity and lower the barriers for developing new experiments and testing more AM research hypothesis.

Furthermore, the implementation of the software in MATLAB is a significant advantage because of the wide adoption of MATLAB as a tool for data analysis in engineering field. Similar systems that are implemented using a low-level programming language (e.g., C or C++)

require nontrivial computer programming skills, which can be discouraging or even prohibitive for researchers that do not have a strong programming background [23]. The software can be easily extended and utilized for other photopolymerization based AM processes (e.g., stereolithography) to advance the process modeling and control for adopting AM (3D printing) in more demanding industrial applications.

Additional information about the software codes and tests can be found at the author's GitHub repository (<https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing>).

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. CMMI-1234561. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation. We thank Harrison Jones, Will Borzon, Changxuan Zhao and Dr. Amit Jariwala for their contributions to the hardware setup and the software development. MATLAB is a registered trademark of The MathWorks, Inc. A full version of the software codes and documents are available in the authors' GitHub repository (link: <https://github.com/Code-XYZxyz/real-time-interferometric-measurement-control-for-photopolymer-additive-manufacturing>).

REFERENCES

- [1] Measurement science roadmap for metal-based additive manufacturing. National Institute of Standards and Technology (NIST) 2013.
- [2] Mani M, Lane BM, M. Alkan Donmez, Feng SC, Moylan SP. A review on measurement science needs for real-time control of additive manufacturing metal powder bed fusion processes. International Journal of Production Research. 2016;1-19.
- [3] Measurement science roadmap for polymer-based additive manufacturing. Gaithersburg MD, USA: National Institute of Standards and Technology; 2016.
- [4] Ly S, Rubenchik AM, Khairallah SA, Guss G, Matthews MJ. Metal vapor micro-jet controls material redistribution in laser powder bed fusion additive manufacturing. Scientific Reports. 2017;7(1).
- [5] Raza I, Iannucci L, Curtis PT. Introducing a multimaterial printer for the deposition of low melting point alloys, elastomer, and ultraviolet curable resin. 3D Printing and Additive Manufacturing. 2017;4(2):83-9.
- [6] Real time monitoring of metal based additive manufacturing; [06-28-2017]. Available from: <http://www.3dprintingprogress.com/articles/11225/real-time-monitoring-of-metal-based-additive-manufacturing>
- [7] Bidare P, Maier RRR, Beck RJ, Shephard JD, Moore AJ. An open-architecture metal powder bed fusion system for in-situ process measurements. Additive Manufacturing. 2017;16:177-85.
- [8] Spears TG, Gold SA. In-process sensing in selective laser melting (slm) additive manufacturing. Integrating Materials and Manufacturing Innovation. 2016;5(2).
- [9] Evertona SK, Hirscha M, Stravroulakisa P, Leacha RK, Clare AT. Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing. Materials and Design. 2016;95(5):431-45.
- [10] Grantham S, Lane B, Neira J, Mekhontsev S, Vlasea M, Hanssen

- L. Optical design and initial results from nist's ammt/temps facility. Proc. SPIE 9738, Laser 3D Manufacturing III, 97380S; 2016.
- [11] Vlasea ML, Lane BM, Lopez FF, Mekhontsev S, Donmez MA. Development of powder bed fusion additive manufacturing test bed for enhanced real time process control. International Solid Freeform Fabrication Symposium. Austin, TX USA; 2015.
- [12] Materialise broadens metal offering and launches inspection software; [cited 2017]. Available from: <http://www.materialise.com/en/press-releases/materialise-broadens-metal-offering-and-launches-inspection-software>
- [13] Jariwala AS. Modeling and process planning for exposure controlled projection lithography [Ph.D. Dissertation]. [Atlanta, USA]: Georgia Institute of Technology; 2013.
- [14] Zhao X, Rosen DW. Real-time interferometric monitoring and measuring of photopolymerization based stereolithographic additive manufacturing process: Sensor model and algorithm. Measurement Science and Technology. 2017 January;28(1).
- [15] Zhao X, Rosen DW. A data mining approach in real-time measurement for polymer additive manufacturing process with exposure controlled projection lithography. Journal of Manufacturing Systems. 2017;Special Issue: Cybermanufacturing.
- [16] Slaev VA, Chunovkina AG, Mironovsky LA. Metrology and theory of measurement: Walter de Gruyter; 2013.
- [17] MathWorks. Perform parallel computations on multicore computers, gpus, and computer clusters. Available from: <https://www.mathworks.com/products/parallel-computing.html>
- [18] MathWorks. Batch (run matlab script or function on worker). Available from: <http://www.mathworks.com/help/distcomp/batch.html>
- [19] Zhao X, Rosen DW. Experimental validation and characterization of a real-time metrology system for photopolymerization based stereolithographic additive manufacturing process. International Journal of Advanced Manufacturing Technology. 2016.
- [20] Zhao X, Rosen DW. Parallel computing enabled real-time interferometric measurement and feedback control for a photopolymer based lithographic additive manufacturing process. Mechatronics. 2017 (Submitted).
- [21] Zhao X. Process measurement and control for exposure controlled projection lithography [Ph.D. Dissertation]. [Atlanta, USA]: Georgia Institute of Technology; 2017. 428 p.
- [22] Wang L, Törngren M, Onori M. Current status and advancement of cyber-physical systems in manufacturing. Journal of Manufacturing Systems. 2015;37:517-27.
- [23] Asaad WF, Santhanam N, McClellan S, Freedman DJ. High-performance execution of psychophysical tasks with complex visual stimuli in matlab. Journal of Neurophysiology. 2013;109(1):249-60.