Science Advances
AAAS

advances.sciencemag.org/cgi/content/full/6/24/eaaz7202/DC1

Supplementary Materials for

Tunable structural color of bottlebrush block copolymers through direct-write 3D printing from solution

Bijal B. Patel, Dylan J. Walsh, Do Hoon Kim, Justin Kwok, Byeongdu Lee, Damien Guironnet, Ying Diao*

*Corresponding author. Email: yingdiao@illinois.edu

Published 10 June 2020, *Sci. Adv.* **6**, eaaz7202 (2020)
DOI: 10.1126/sciadv.aaz7202

The PDF file includes:

Other Supplementary Material for this manuscript includes the following:

(available at advances.sciencemag.org/cgi/content/full/6/24/eaaz7202/DC1)

§1. Description of Supplemental Videos

Supplemental Video 1: Microscope camera video of drying dropcast samples (**Figure 1F)**. Samples dropcast from 40 µL of 100 mg/mL solution in THF onto plasma-cleaned bare silicon wafers at room temperature. Droplets are ~ 6 mm in diameter.

Supplemental Video 2: 5x speed pen camera video of the meniscus during printing of a meanderline pattern at a speed of 60 mm/min, substrate temperature of 50°C, and applied pressure of 30 kPa. Printed film is shown in the 3rd/4th rows and 3rd column of **Figure 2C** (lower panel) in the main text.

Supplemental Video 3: 10x speed pen camera video of the meniscus during printing of the 50°C, constant condition chameleon shown in **Figure 2D** of the main text.

Supplemental Video 4: Cellphone camera video clip of layered chameleon (**Figure 2E)** printing onto a bare silicon 4" wafer.

Supplemental Video 5: Mosaic of side-view (transmission) videos analyzed for in situ optical measurements. Printing speeds are noted in the video. The syringe needle shown in the videos has outer diameter ~240 µm.

Supplemental Video 6: Top view videos of the type analyzed for *in situ* optical measurements. Printing speeds are noted in the video. The syringe needle shown in the videos has outer diameter ~240 µm.

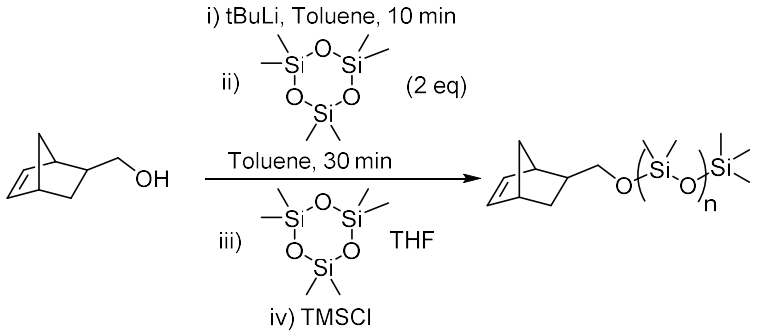
§2. Synthesis and Chemical Characterization

All reactions were performed in an argon-filled glovebox (O_2 < 2 ppm, H_2O < 0.5 ppm) at room temperature using oven dried glassware. THF and toluene were dried using a commercial solvent purification system. rac-Lactide {Aldrich}, and tert-butyllithium solution {1.7 M in pentane, Aldrich} was used as received. 1,8-diazabicyclo[5.4.0]undec-7-ene (DBU) {Aldrich} and hexamethylcyclotrisiloxane {Aldrich, 98%}, was distilled prior to use. Chlorotrimethylsilane (TMSCl) was distilled over CaH_2 and storage under argon. [$(H_2IMes)(3\text{-Br-py})_2(Cl)_2Ru{=}CHPh$], G3 was synthesized according to literature.[48] 5-Norbornene-2-methanol was synthesized according to literature (mixture of 20% exo/endo used)[52].

Gel Permeation Chromatography (GPC) was performed using a Tosoh Ecosec HLC-8320GPC at 40 ºC fitted with a reference column (6.0 mm ID x 15 cm), a guard column (6.0 mm ID x 4.0 cm x 5 µm), and two analytical columns (7.8 mm ID x 30 cm x 5 µm). The reference flow rate is 0.5 mL min^{-1} while the analytical column is at 1.0 mL·min^{-1}. THF (HPLC grade) was used as the eluent, and polystyrene standards (15 points ranging from 500 Mw to 8.42 million Mw) were used as the general calibration. An additional calibration was created for specifically for linear polylactic acid and only used for linear polylactic acid (10 points ranging from 500 Mw to 10,000 Mw).

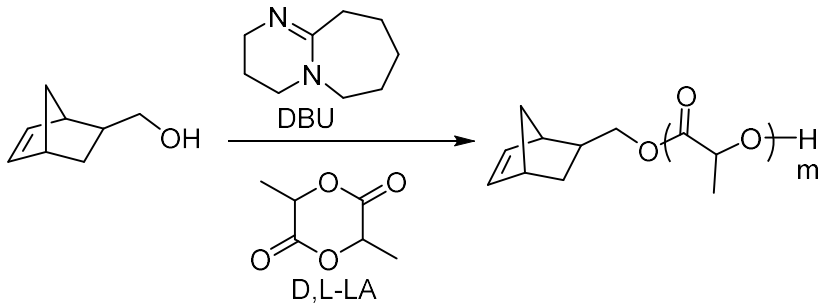
Triple Detection Gel Permeation Chromatography (t-GPC) was performed using a Viscotek GPCmax pump and TDA302 triple detector (Refractive Index, 90º and 7º light scattering, Viscometer) at 35 ºC fitted with two mixed-bed analytical columns (PolyAnalytik PAS-M: 8 mmID x 30 cm length, 10 µm particles, exclusion limit 20,000,000 Da relative to polystyrene). The flowrate was 1.0 mL·min^{-1}. THF (HPLC grade) was used as the eluent. The detectors were calibrated with a narrow polystyrene standard (Mw= 99,000 Da). Performed by PolyAnalytik Inc. (London, Canada).

Procedure for the synthesis of PDMS macromonomers:



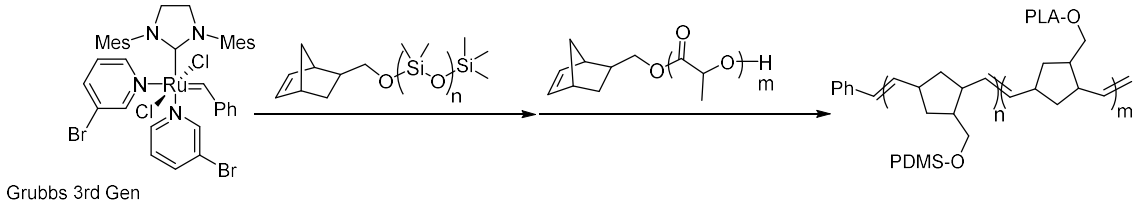
Procedure for the anionic ring opening polymerization (ROP) of siloxane has previously been described by our group, and re-described here.[30] M_n(GPC with respect to PS standards)= 6,200 g/mol; M_w/M_n=1.05 (GPC traces in **Figure S4).**

Procedure for the synthesis of PLA macromonomers



Procedure for the ring opening polymerization (ROP) of lactide has previously been described by our group, and re-described here.[15,30] M_n(GPC with respect to PLA standards)= 5,100 g/mol; M_w/M_n=1.05 (GPC traces in **Figure S4Figure S2**).

Procedure to obtain kinetic data for the synthesis of PDMS-b-PLA bottlebrush polymers



In an oven-dried 20 mL glass vial, PDMS macromonomer (300 mg, 0.0484 mmol) was dissolved into THF (3 ml). The polymerization is initiated by adding G3 via a stock solution (0.5 ml add of: 4.28 mg G3 in 2.5 ml THF). Time points were obtained by taken 70 µL aliquots and injecting into 1 ml of THF with a large excess of ethyl vinyl ether. Each aliquot was analyzed by GPC. After 30 min, PLA macromonomer (247 mg, 0.0484 mmol) was added in THF (3 ml). Time points were collected and analyzed by GPC during this reaction period as well. GPC traces shown in **Figures S1, S2**. Analyzed data shown in **Figure S3**.

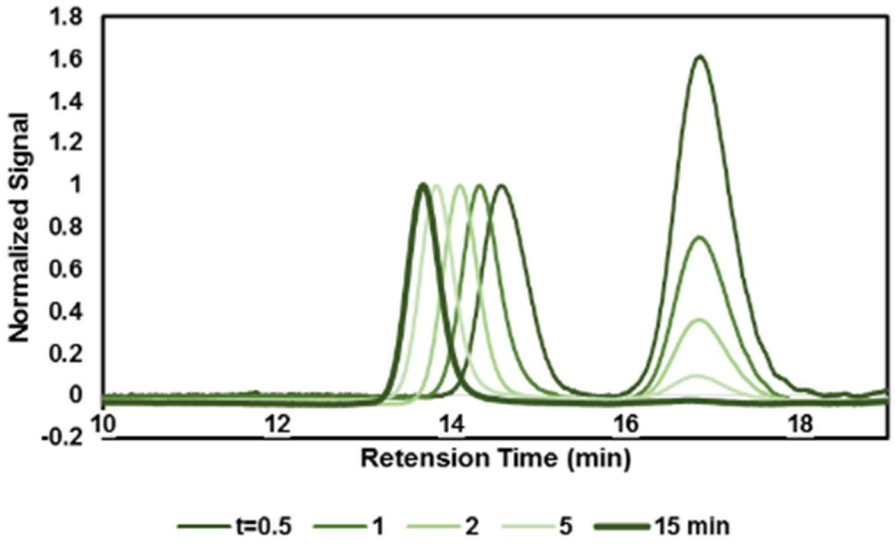


Figure S1. GPC traces for the ROMP of PDMS macromonomers

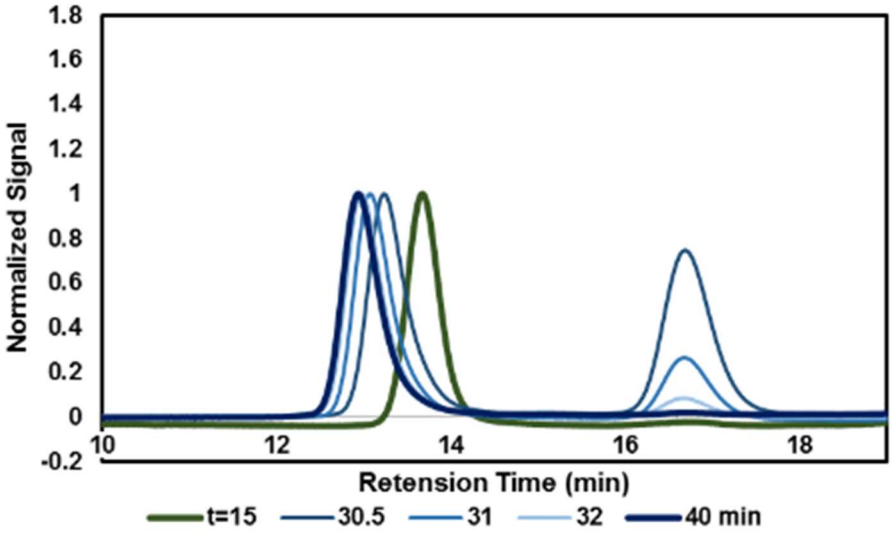


Figure S2. GPC traces for the ROMP of PLA macromonomers continuing off of the PDMS bottlebrush. The bold green trace is the PDMS bottlebrush.

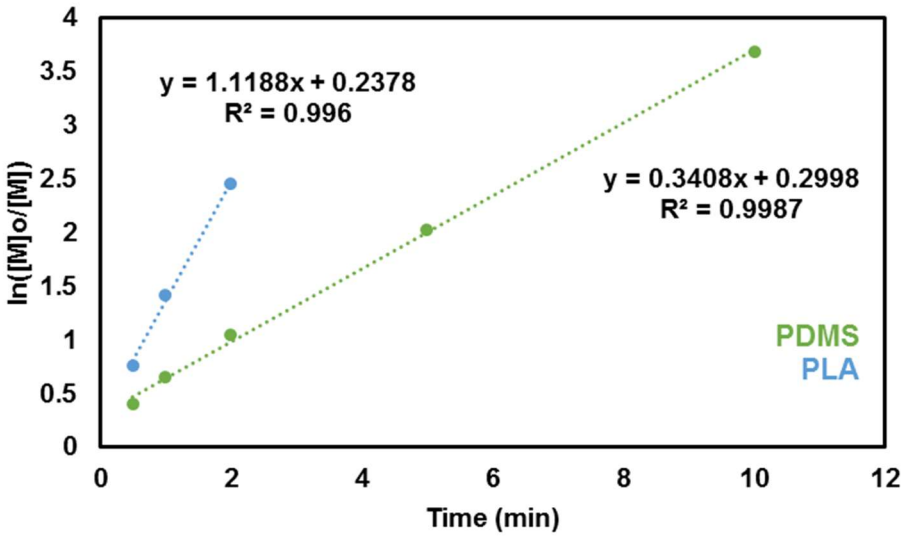
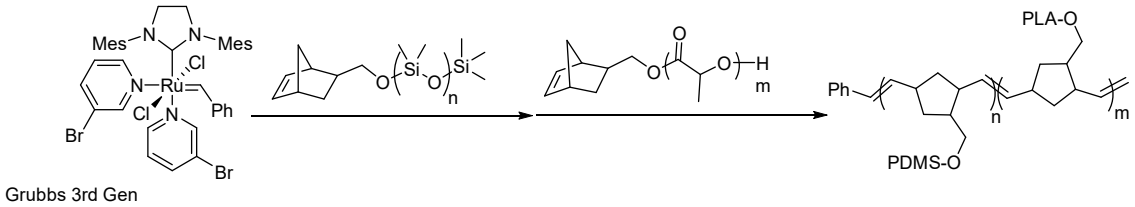


Figure S3. Kinetic plot for the synthesis of PDMS-b-PLA bottlebrush..

Procedure to obtain kinetic data for the synthesis of PDMS-b-PLA bottlebrush polymers



In an oven-dried round bottom flask, PDMS macromonomer (3 g, 0.484 mmol) was dissolved into THF (30 ml). The polymerization is initiated by adding G3 via a stock solution (5 ml add of: 4.28 mg in 10 ml THF). Each aliquots was analyzed by GPC. After 30 min, PLA macromonomer (2.47 g, 0.484 mmol) was added in THF (30 ml). After 30 min, a large excess of ethyl vinyl ether (large excess with respect to [Ru]) was added to the reaction mixture then poured into cold methanol (-30 ℃) and a centrifuge was used to isolate the resulting polymer. The polymer was dried under vacuum and then analyzed by GPC (**Figure S4)**. M_n(t-GPC)= 1,930,000 g/mol; M_w/M_n=1.09

(2 % of unfunctionalized (no norbornene end group) brush is present by GPC)

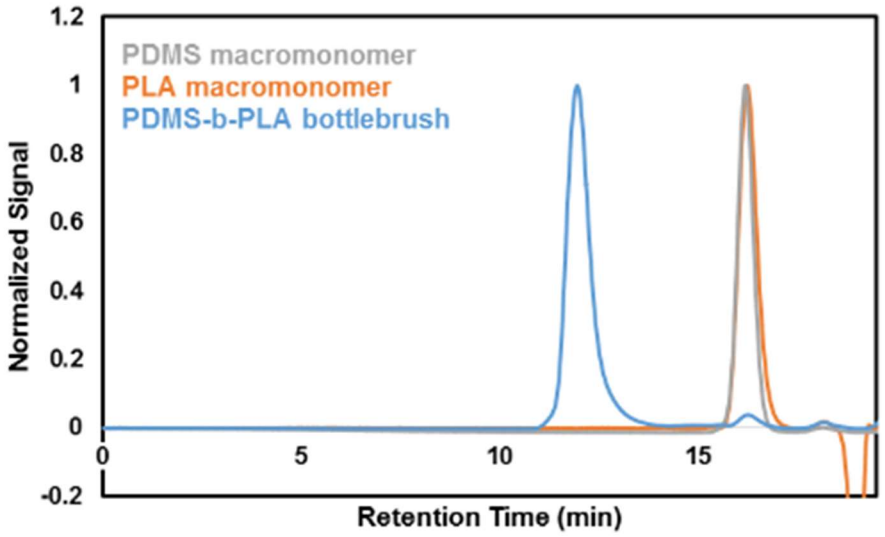


Figure S4: GPC traces of PDMS-b-PLA bottlebrush and PDMS/PLA macromonomers.

§3. Thermal Properties (Differential Scanning Calorimetry)

The as-prepared bottlebrush block copolymer reflected the glass transition of the constituent arm chemistries very close to the homopolymer brush and homopolymer arm transition temperatures as summarized below. (Excerpts from 2nd heats plotted).

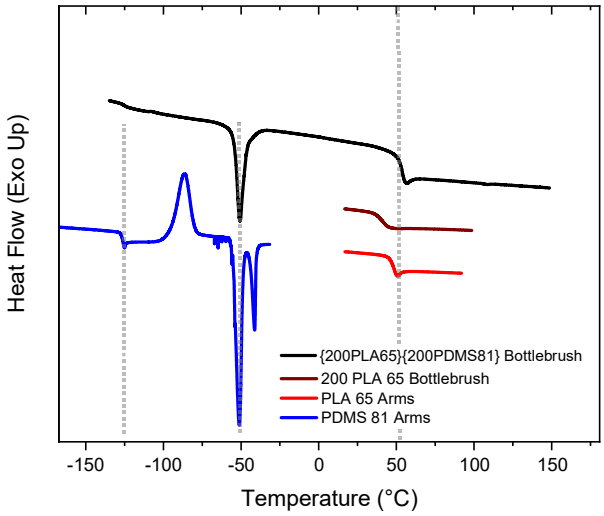


Figure S5. Replotted features excerpted from DSC curves for bottlebrush block copolymer, homopolymer bottlebrush, and linear polymer arms.

Measurement Parameters.

Samples of ~5 mg were measured into DSC Consumables Aluminum Tzero pans and scanned on a TA instruments Discovery 2500 Differential Scanning Calorimeter (DSC) at the University of Illinois Materials Research Laboratory. Samples were first equilibrated at -180°C. All temperature ramps were performed at 10°C/min with 2 minute isotherms at endpoints.

Table S1. Thermal Properties of synthesized Bottlebrush

Parameter	Measured from 2nd heat	Literature Value (linear polymer)
PDMS T_g	-125.51 C	-123.15 C (150 K)[33]
PDMS T_m	-50.90 C	-53 C (220 K)[33]
PLA T_g	53.19 C	49.9 C - 56.9 C (323K-330K)[33]

§4. Small-Angle X-ray Scattering and Scanning Electron Microscopy Analysis of BBCP ink

Experimental Parameters:

Experiments were performed at beamline 12-ID-B of the Advanced Photon Source (Lemont, IL) with beam energy 13.3 keV using the Pilatus 2M 2D detector at a sample-detector distance of 3.6 m. Q-calibration was performed against a silver behenate standard. 2D data reduction was performed using the Nika package for Igor Pro (WaveMetrics, Lake Oswego, OR, USA) developed by Jan Ilavsky.[51]

Ink Characterization (10.1 wt% solution)

Solutions of the 10.1 wt% (100 mg/mL) BBCP in THF were prepared via dissolution with stirring at room temperature, followed by aging for 2 hours without stirring and exhibited a faint purple reflection. Solutions were loaded into 1.0 mm, thin-walled quartz capillaries (Charles Supper Company, Inc. – Natick, MA). For each image/profile shown below, 10 scans were taken at 0.1s exposure times and averaged. Azimuthal averaging was performed from 0 to 180 degrees.

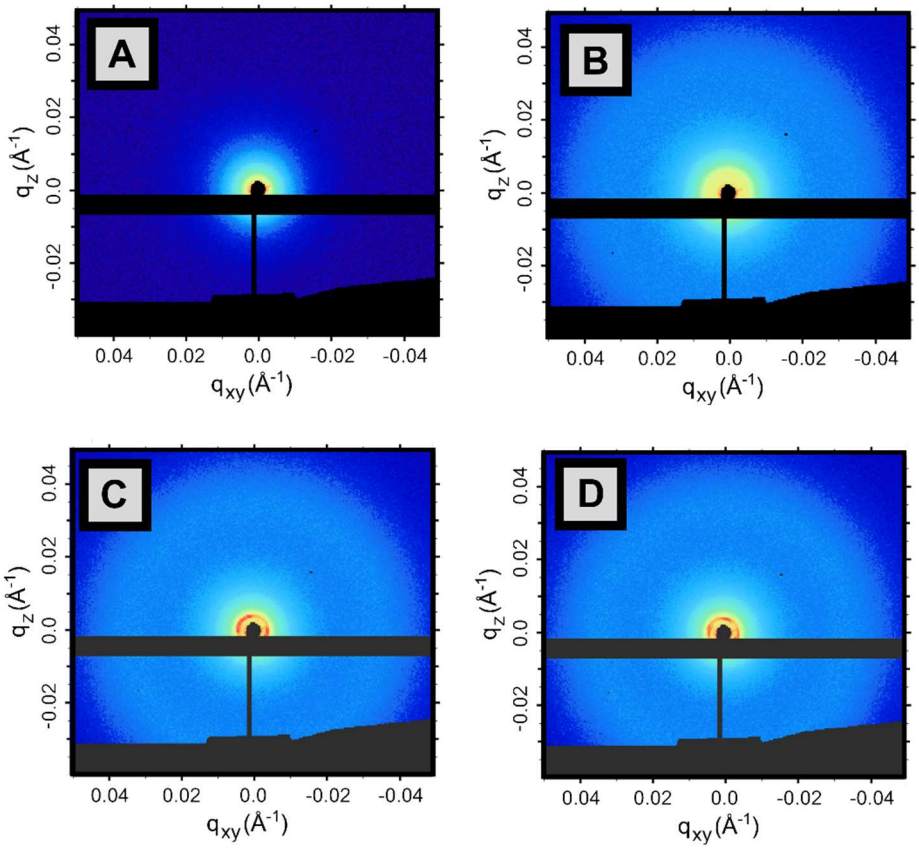


Figure S6. Raw 2D SAXS data for solutions in capillaries. (A) Pure THF, (B) 10.1 wt% PDMS-b-PLA with backbone DP of 200 repeat units, (C) and (D) two trials of 10.1 wt% PDMS-b-PLA with backbone DP of 400 repeat units.

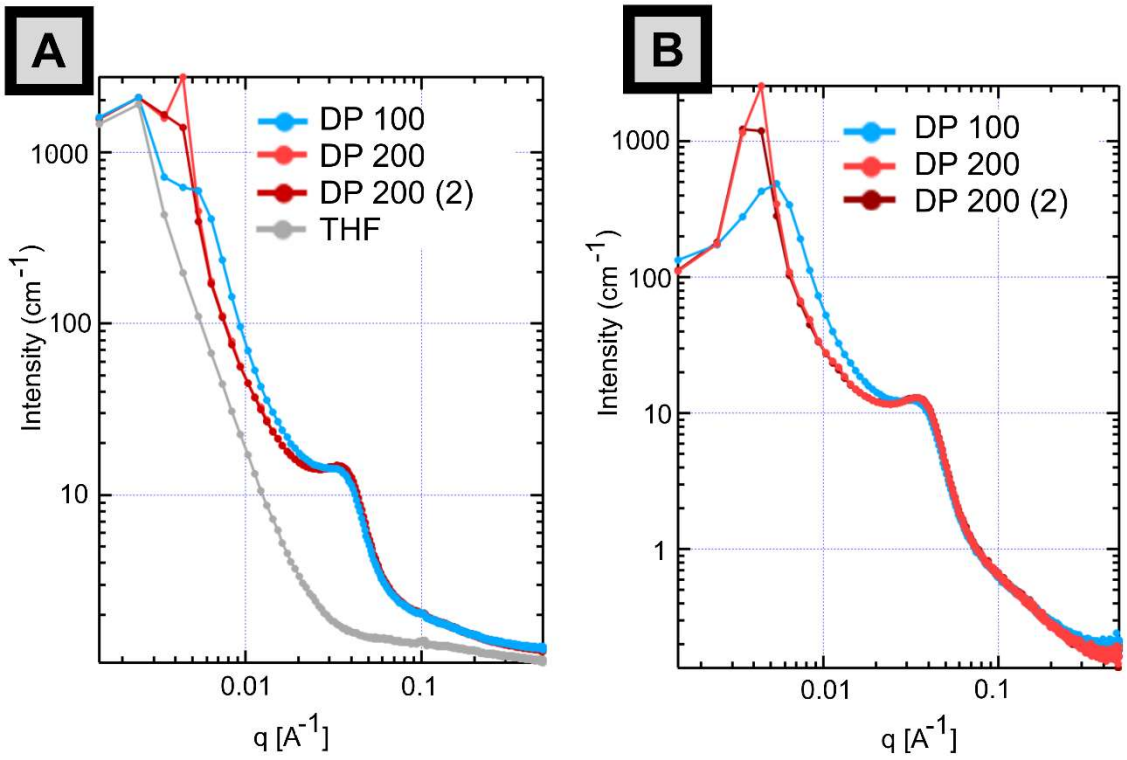


Figure S7. Azimuthally averaged profiles for capillary SAXS measurements (a) before and (b) after subtraction of pure solvent background.

Scanning Electron Microscopy:

To further support this conclusion, we have performed scanning electron microscopy of a solution of 100 mg/mL BBCP in THF prepared via a freeze drying method. In brief: a small quantity of solution was dropped onto a silicon wafer and quickly covered by a glass slide and immersed in liquid nitrogen to freeze the solvent. The slide was separated from the wafer while still immersed and immediately placed into an evacuated microscope stage held at -150 C. While under vacuum, the temperature was slowly (~1C/min) increased to allow solvent to melt and evaporate. The resulting film was coated with 5 nm Au/Pd and imaged at 7 keV in a JEOL 7000F Analytical SEM.

We obtain the following micrograph:

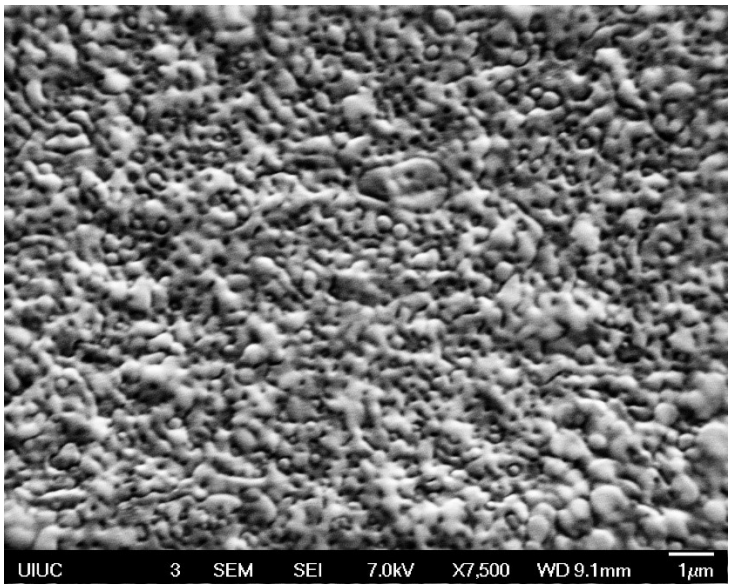


Figure S8. SEM micrograph of freeze-dried 100 mg/mL BBCP solution.

The freeze-dried film comprises a large number of partially-fused spherical micelles. Performing a simple image analysis by measurement of the diameter of 50 isolated particles yields an average diameter of 190 ± 45 nm, in rough agreement particle size measured from SAXS (156.7 nm).

§5. 3D printer Hardware and Software

Parts and software designed by BP during the course of this work are provided in the supporting information in the zipped folder "SI_HardwareSoftware". Latest versions of the printer control software (PolyChemPrint) written for this work can be found at the following GitHub link: https://github.com/BijalBPatel/PolyChemPrint3#polychemprint3 and at http://diao.scs.illinois.edu/Diao_Lab. All code written for this work is released under the Open Source UIUC/NCSA License. Software from other sources used in this work is listed below.

Lulzbot Taz 6 3D Printer Hardware modifications

The Lulzbot Taz 6 3D Printer was purchased from Aleph Objects, Inc. The original thermoplastic print head was removed from its mount but left connected to avoid a temperature probe error. A new syringe mount was custom modeled in Autodesk Inventor and 3D printed with a simple grid pattern of holes for supporting the pen camera for monitoring nozzle position. Holes were enlarged with a soldering iron and threaded using ¼-20 socket head cap screws (used to secure the camera). The syringe was secured to the mount with two zip ties during printing. The original PEI print bed was protected from solvent by a large ¼" thick glass plate secured with two medium binder clips. The filament reel and original hot end scrubbing pad were removed.

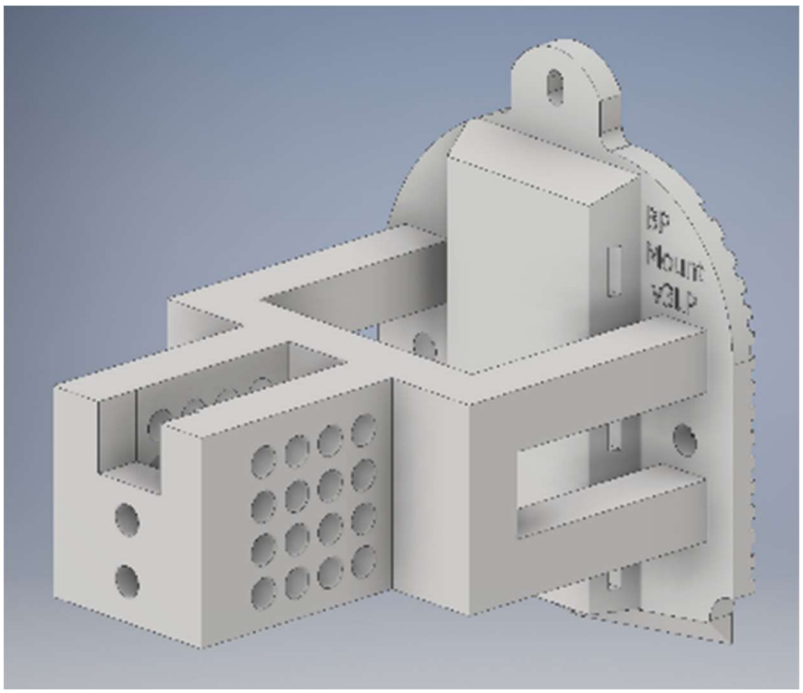


Figure S9. Custom syringe mount, CAD and STL files available for download in supplemental zipped folder.

Lulzbot Taz 6 Firmware modifications

Modifications to the open source Marlin firmware on the Taz 6 (https://github.com/alephobjects/Marlin, https://github.com/MarlinFirmware/Marlin) were made using the open source arduinoIDE 1.8.9 (https://www.arduino.cc/en/main/software). Three major changes were made to the stock Taz 6 Marlin firmware (original sources). First, the baud rate was changed to 115200. Second, the motion buffer was reduced to 1 command to prevent

queueing of motion commands. Third, Marlins "ok…" command receipt response was suppressed until the end of the current motion step in order to signal to PolyChemPrint that motion was complete for synchronization with the extruder. The latest version of the modified firmware source and compiled .hex file are available for download from the zipped folder or the link at the beginning of this section.

Pneumatic Extrusion Hardware

The Ultimus V dispensing system was procured from Nordson EFD, Inc. The unit was used without modification along with 3 mL polypropylene amber syringe barrels and 32 Gauge standard needle tips. Standard white polyethylene pistons were used to avoid excess evaporation of solvent. In order to evaluate the chemical resistance of the barrel, nozzles, and piston, each was left in a sealed (parafilmed) vial of tetrahydrofuran for 2 weeks before a sample of the solvent was taken and dropcast onto clean silicon. No residue was observed and no visible degradation of the parts was seen. Nevertheless, parts were only used once and were in contact with the solvent a maximum of ~7 hours for each set of experiments. It was found that over time (~weeks) the O-ring at the seat of the low-pressure syringe adapter began to crack due to solvent exposure and lead to uneven dispensing. All data presented in this manuscript was collected with freshly installed o-rings.

PolyChemPrint Software

A custom software program was written to control the Ultimus V extruder (over a RS-232 serial port) and the Taz 6 (through USB). The current version of the program (v2.2) was written in Perl and runs via a terminal interface on Linux Debian 9/10. In brief, the program has three modes of operation: (1) direct hardware control (line-by-line command execution), (2) pre-programmed, customizable, scripts for simple patterns such as meander-lines, and (3) importing of GCODE files. As of version 2.2, imported GCODE files are only used for motion paths, while print speed, pressure, travel speed, z-hop height, and z-speed are fully customizable within PolyChemPrint. Automatic data logging is supported. Latest versions of the software and fully documented source files are available at the link at the beginning of this section. Version 2.2 is available for download from the supplemental zipped folder. A brief software overview (through screenshots) is provided in the following figures.

The authors thank John Roshek at the School of Chemical Sciences Electronics shop for invaluable advice early in the process.

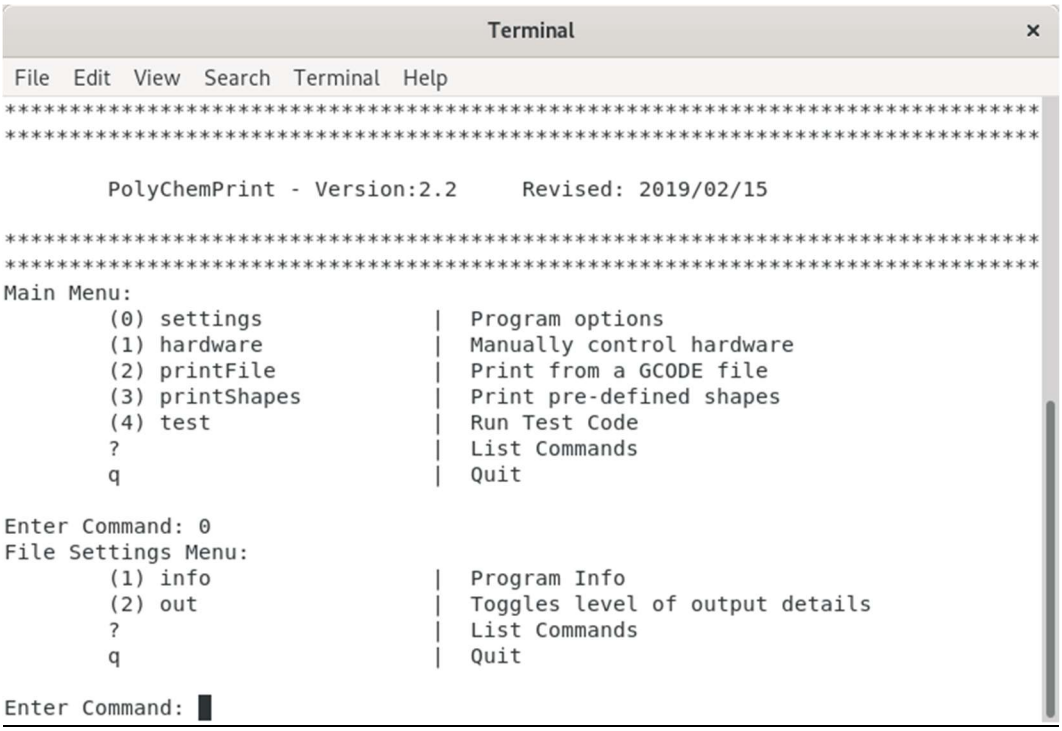


Figure S10. Main Menu and Settings

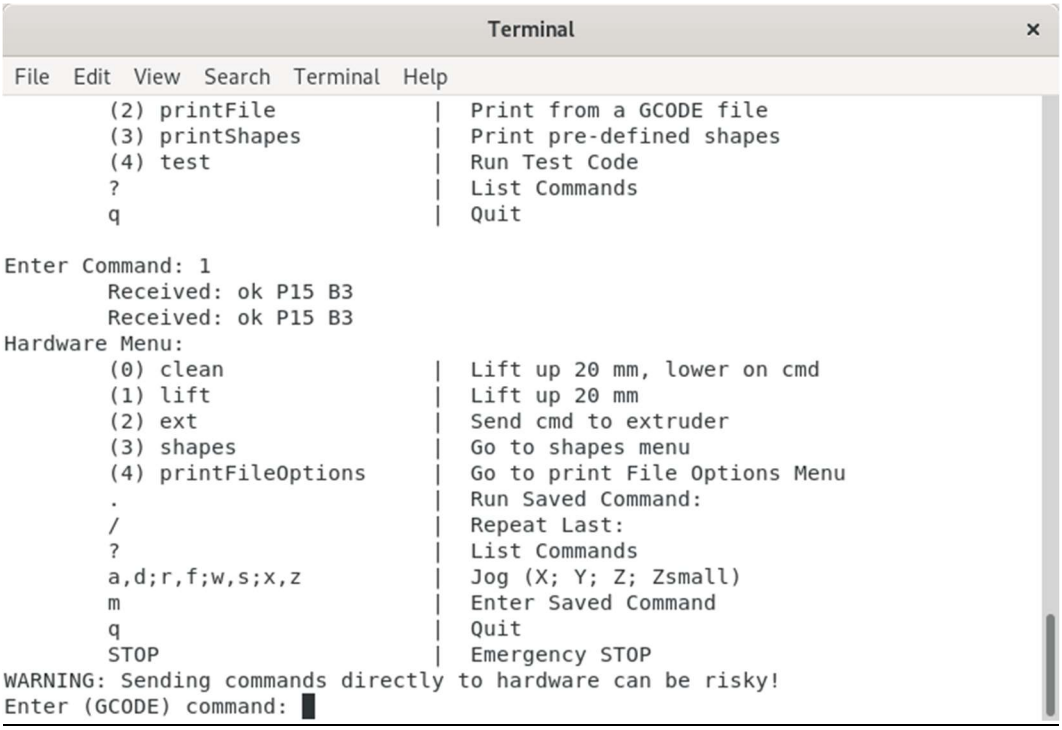


Figure S11. Hardware Menu

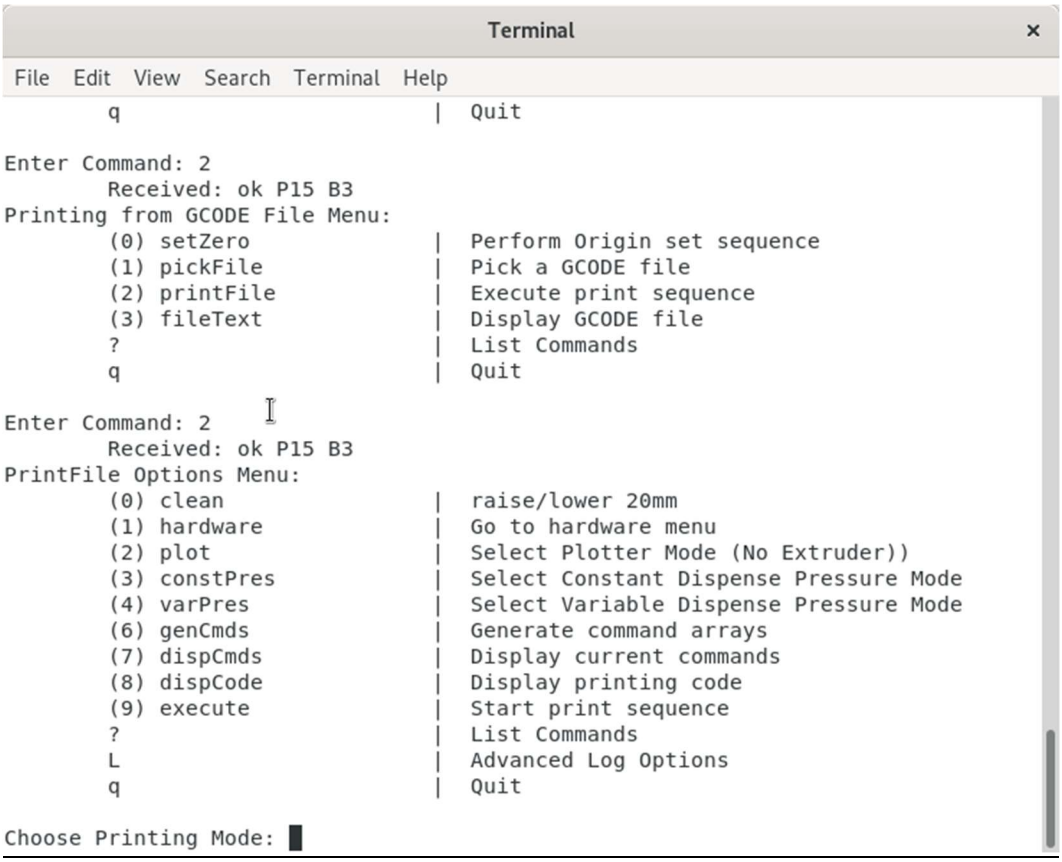


Figure S12. Printing from GCode File Menu

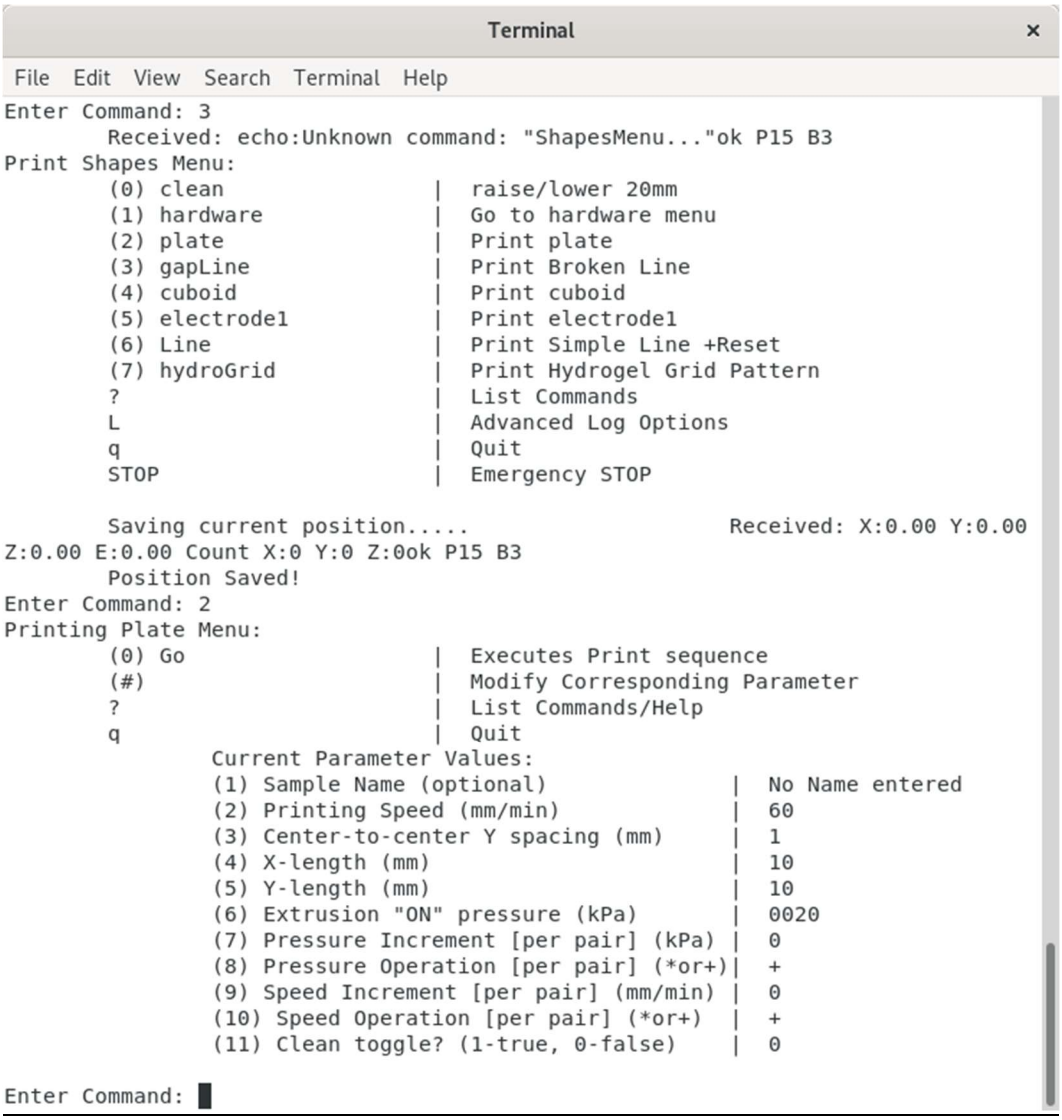


Figure S13. Printing Parameterized Shape Menu

§6. Printing Reproducibility Test

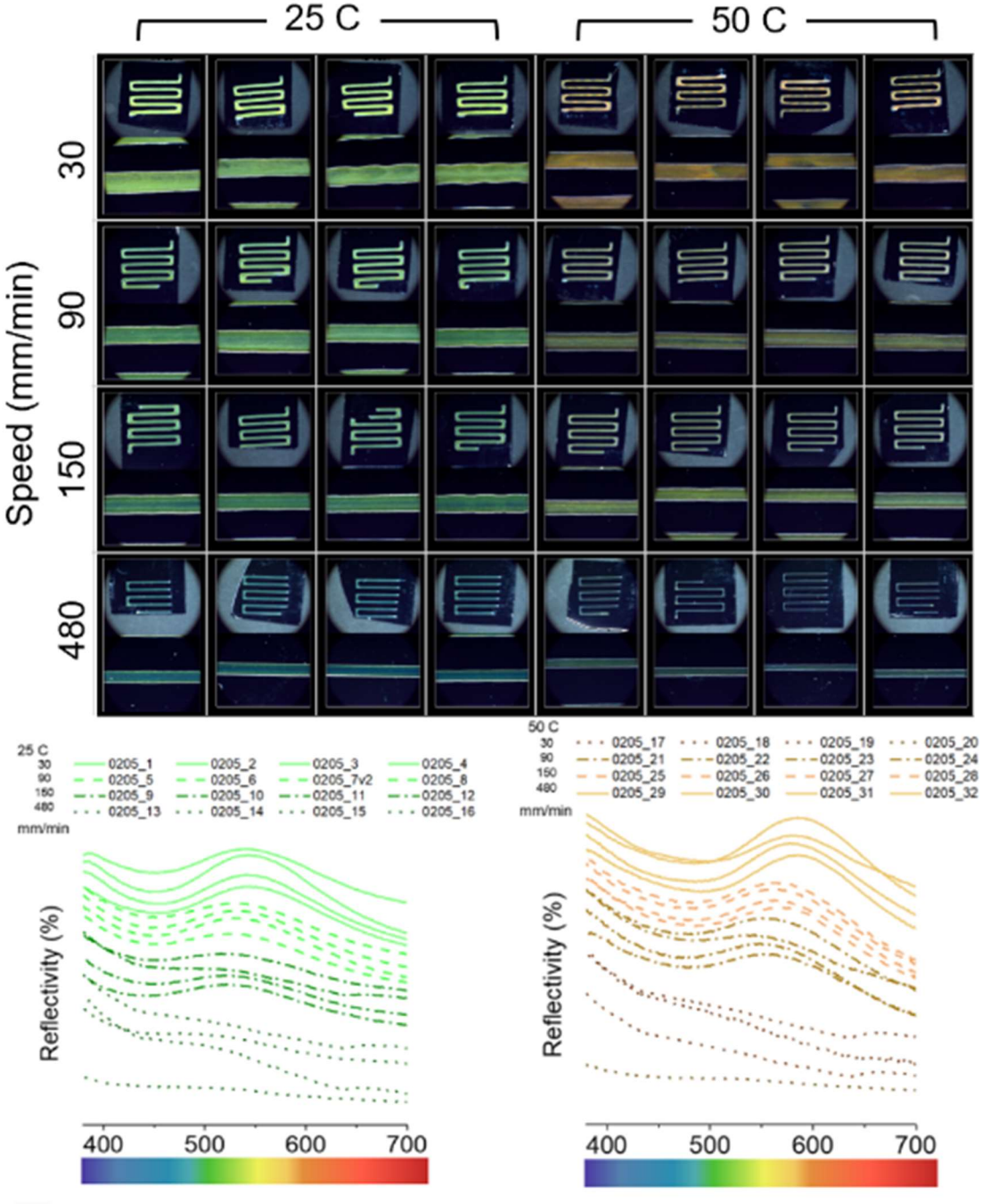


Figure S14. Microscope images and UV-Vis diffuse reflection spectra a series of samples printed repeatedly at the same conditions.

§7. Demonstration Prints

In order to print the complex chameleon patterns, a vector drawing was first drawn in Inkscape, a free vector editing software. Then, the sections of the drawing at each print bed temperature were separated into separate image files. Images were converted to GCode using the free GCodeTools plugin for inkscape with a pen diameter of 10 mm, minimum arc radius of 1mm, depth function of 1, Z safe height of 3 mm, print speed of 9999, penetration speed of 9998, and passing speed of 1000 [replaceable values within PolyChemPrint]. This scheme is quite easy to use and allowed the authors to create the patterns from scratch in a matter of hours.

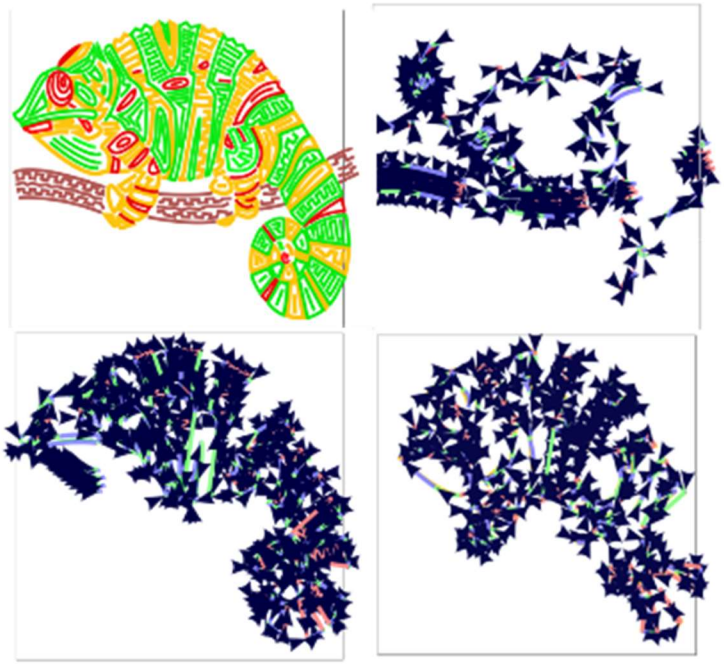


Figure S15. Top left: Original inkscape chameleon drawing and (clockwise): 65 C layer, 50 C layer, and 25 C layer with motion vectors overlayed.

Chameleon patterns were printed onto bare 100mm silicon wafers (University Wafers No.1113) and imaged on the Keyence VK-X1000 3D Laser Scanning Confocal microscope in image acquisition mode with the full ring light enabled. ~150 individual images each were taken and manually stitched together using Paint.Net, a free lightweight photo editor, to obtain the originals below. Images shown below are each stitched from ~100 optical microscopy images under diffuse, ring lighting. Dust particles and the faint reflection of the light source were removed from the background for the images in the main text. Images shown below are unedited.

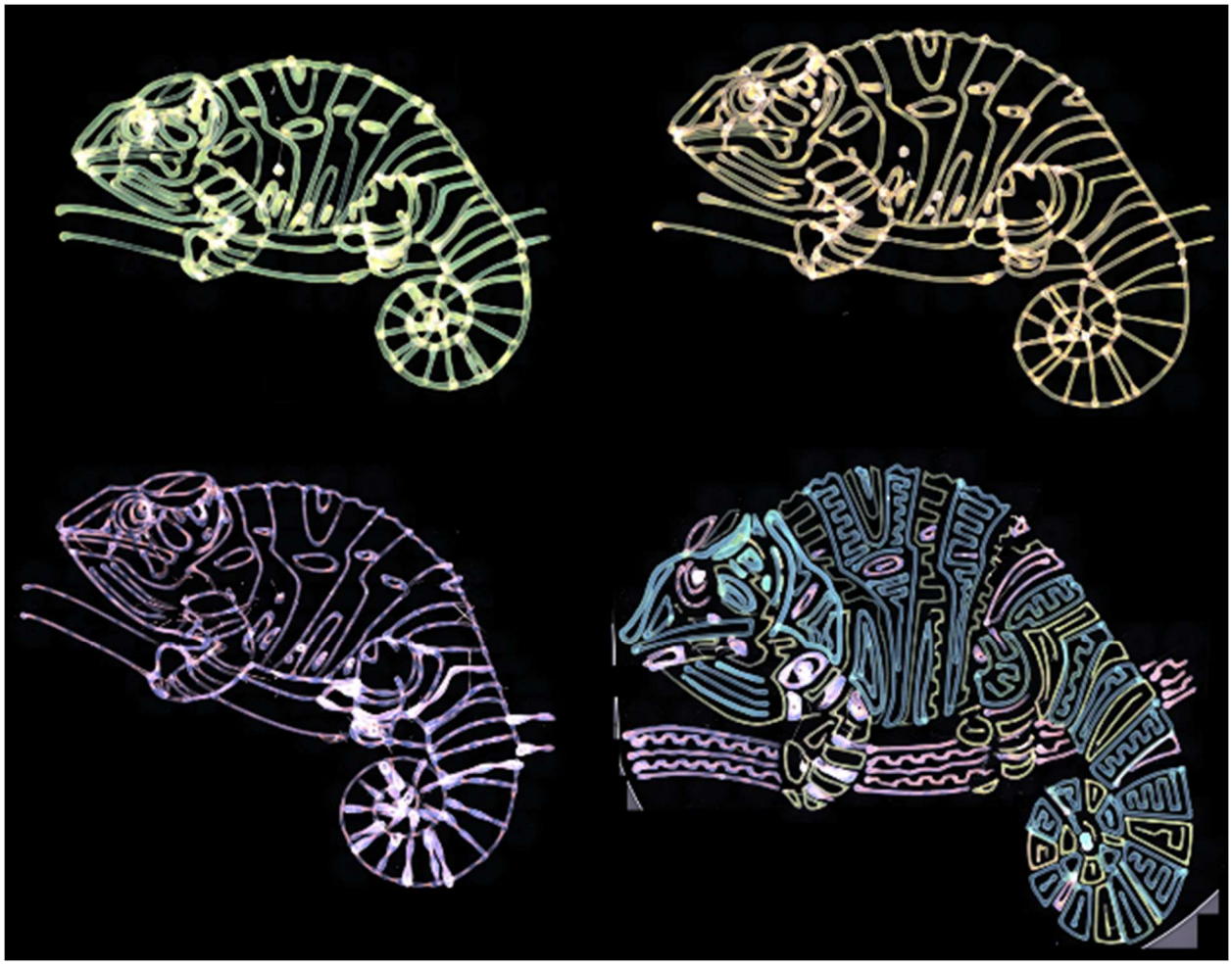


Figure S16. Original stitched microscope images of chameleons printed onto bare Si wafers.

For printing on a curved surface, a 10 mL glass round bottom flask was used as the substrate (Figure S17A). The previous method using inkscape is unsuitable for 3D surfaces, so the surface was modeled as a spherical cap in Cura with a very large line thickness (to give spacing between the lines). Printing speed and pressure were held constant for each two prints, with the flask held at 25C (Figure S17B,C,D) and ~50C (Figure S17 E,F,G). To improve the visibility of the lines, the flask is imaged both empty, and filled with a dark ink to provide an absorbing background. Image under a microscope at normal incidence reveals that there is a redshift in the reflected wavelength with increasing substrate temperature, consistent with the results on planar surfaces.

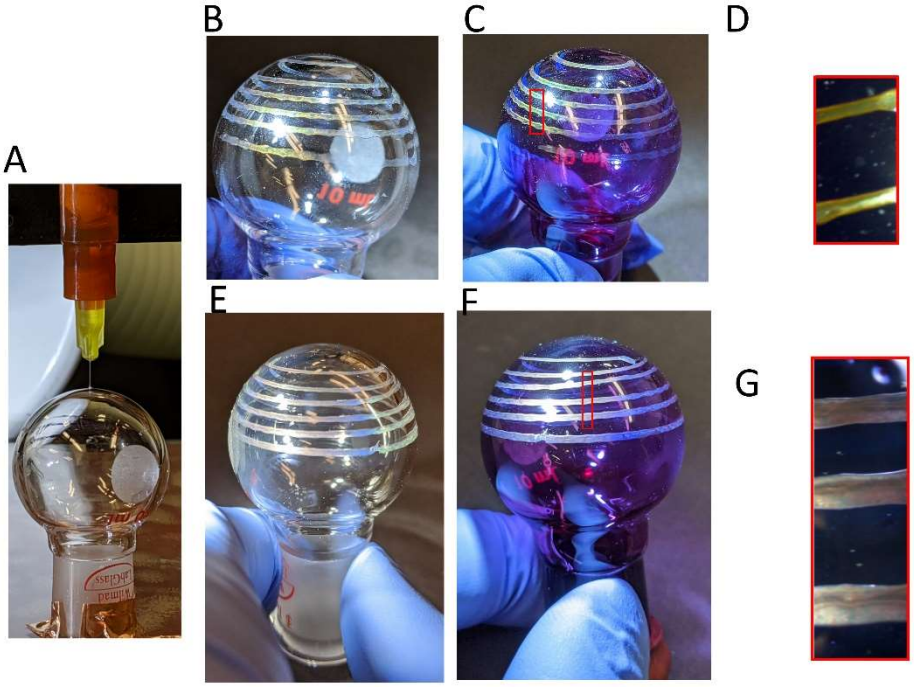


Figure S17. Printing on a curved surface. **(A)** Image of the printer at the start of the print. **(B – D)** Images of an example print at ambient temperature (25C) with an empty round bottom flask (transmissive background), dark ink within the flask (absorbing background), and under a microscope camera, respectively. Upper image in red outline is a magnification of the lower image. **(E – G)** Images of an example print at elevated temperature (T_{glass}~ 50 C). Photo Credit: Bijal Patel, University of Illinois.

§8. UV-Vis Diffuse Reflectance Measurements

UV-Vis measurements were taken on a Varian Cary 5G spectrophotometer using the internal diffuse reflection accessory at the Illinois Materials Research Laboratory. Samples were taken in diffuse reflection mode to eliminate the (mostly) specular reflection signal from the bare silicon substrate. Measurements were taken with Zero/Baseline correction referenced against the empty chamber and Spectralon diffuse reflectance standard, respectively. Signal to Noise Resolution (SNR) mode was used with a threshold of 50:1 and a timeout of 0.25s.

Raw spectra for the large dataset in **Figure 3A** are reproduced below.

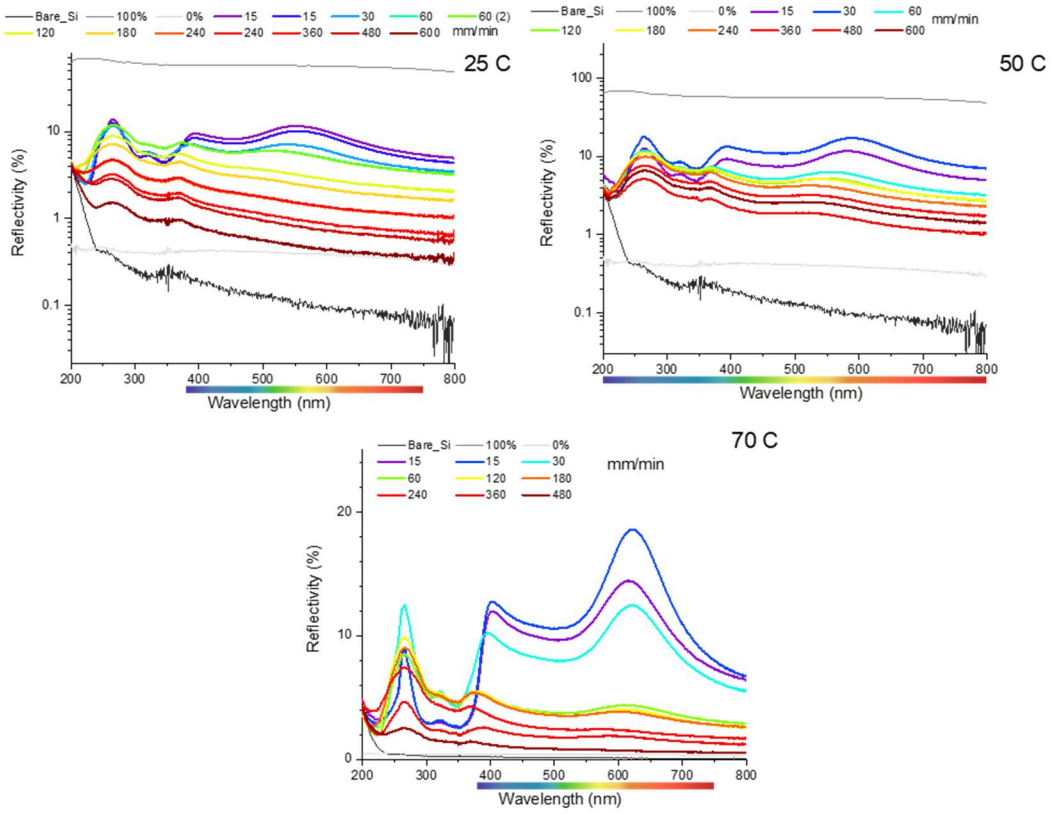


Figure S18. Raw UV-Vis reflection spectra for printed samples shown in **Figure 2C** of the main text.

Peaks were fit to Lorentzian functions in Origin Pro with the initial starting point for iteration placed manually (Aside from the 25 C very thin films, rescaling the y data and restricting the domain to ~450 – 750 nm made the peaks very obvious). In nearly all cases the location of the second order peak ($\lambda_2 = \lambda_1/2$) was missing in the obtained spectra. When deconvoluting the spectra, the data to the left of the n=2 peak that was visible was cropped out and the remaining decay was fit as its own Lorentzian peak.

§9. Volume Fraction Estimation

The volume fraction of PDMS ($\Phi_{PDMS} = 0.61$) was found by substituting the following data

Table S2. Synthetic and chemical properties of polymacromonomers

Species	Mn(g/mol)	DP	$\rho\left(\frac{g}{cm^3}\right)$	Density Citation
PLA-NB	5100	200	1.25	34
PDMS-NB	6200	200	0.965	33

into the following equation, as per Dalsin et al.[9]

$$\phi_{PDMS} = \frac{(M_{n,PDMS-NB} * DP_{PDMS-NB}/\rho_{PDMS-NB})}{(M_{n,PDMS-NB} * DP_{PDMS-NB}/\rho_{PDMS-N}\ )) + (M_{n,PLA-NB} * DP_{PL}\quad /\rho_{PLA-}\quad))}$$

Here, we approximate the density of the pdms-NB macromonomer to be the same as the bulk value of the PDMS/PLA constituent.

§10. Scanning Electron Micrographs of Printed Films

Preparation of Cross-Sections

It was quite challenging to generate undamaged cross-sections of printed films, we suspect due to the very low glass transition of PDMS. The following approach was settled on after much trial and error with simpler methods of freeze fracture and was found to provide acceptable yield (~50%). Alternatively, ultramicrotome is an effective – albeit time-consuming method for sectioning films (thank you to Dr. Scott Robinson at the Beckman Institute at the University of Illinois).

In our scheme, a "double boiler" setup of 1 small crystallization dish was placed inside a larger recrystallization dish resting on a polystyrene board. Liquid nitrogen was poured into both vessels and allowed to equilibrate after the first boil. Samples printed on silicon wafers were carefully delaminated from the silicon with a new, clean steel razorblade. The lower half of each section was sandwiched between two pieces of copper tape and dropped into the LN2 bath and left for 25 minutes. Meanwhile, two pairs of stainless-steel forceps (with insulation taped onto the handles) were placed into the 'outer' crystallization dish to chill them. After 25 minutes, the copper tape sample was picked up with 1 set of forceps and within the LN2 bath, the other set of forceps was scraped across the projecting sample to fracture. Then, the copper tape was separated and replaced lower down to expose ~0.5cm of polymer leading up to the edge for good coating of Au/Pd in the next step.

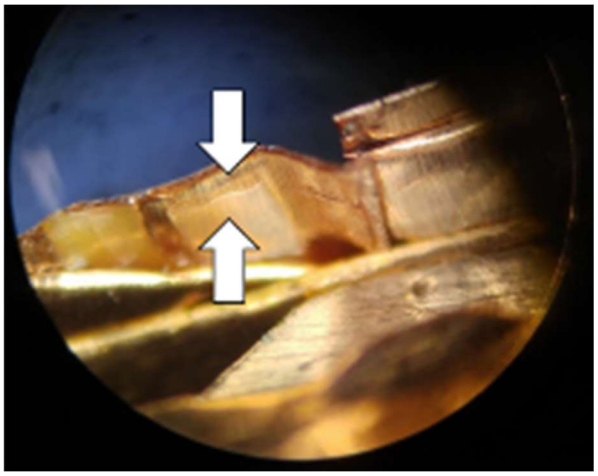


Figure S19. Fractured polymer thin-film sample prior to AuPd coating.

To prevent charging during SEM, samples were coated for 40 seconds in the Emitech K575 with Au/Pd at 20 mA (~2.5 Å/s) for a total thickness of ~10 nm. Samples were first mounted in the 'mini-vise' holder before coating.

Scanning Electron Microscopy

Samples were scanned on a JEOL 7000F Analytical SEM at the Illinois Materials Research Laboratory. It was found that the best balance between domain contrast and resolution was obtained by scanning at 3-5 keV at the highest probe current. Samples were found to show adequate contrast without the use of an additional chemical stain. Images that follow are contrast enhanced in ImageJ 2 for clarity.

Dropcast Samples – 25°C

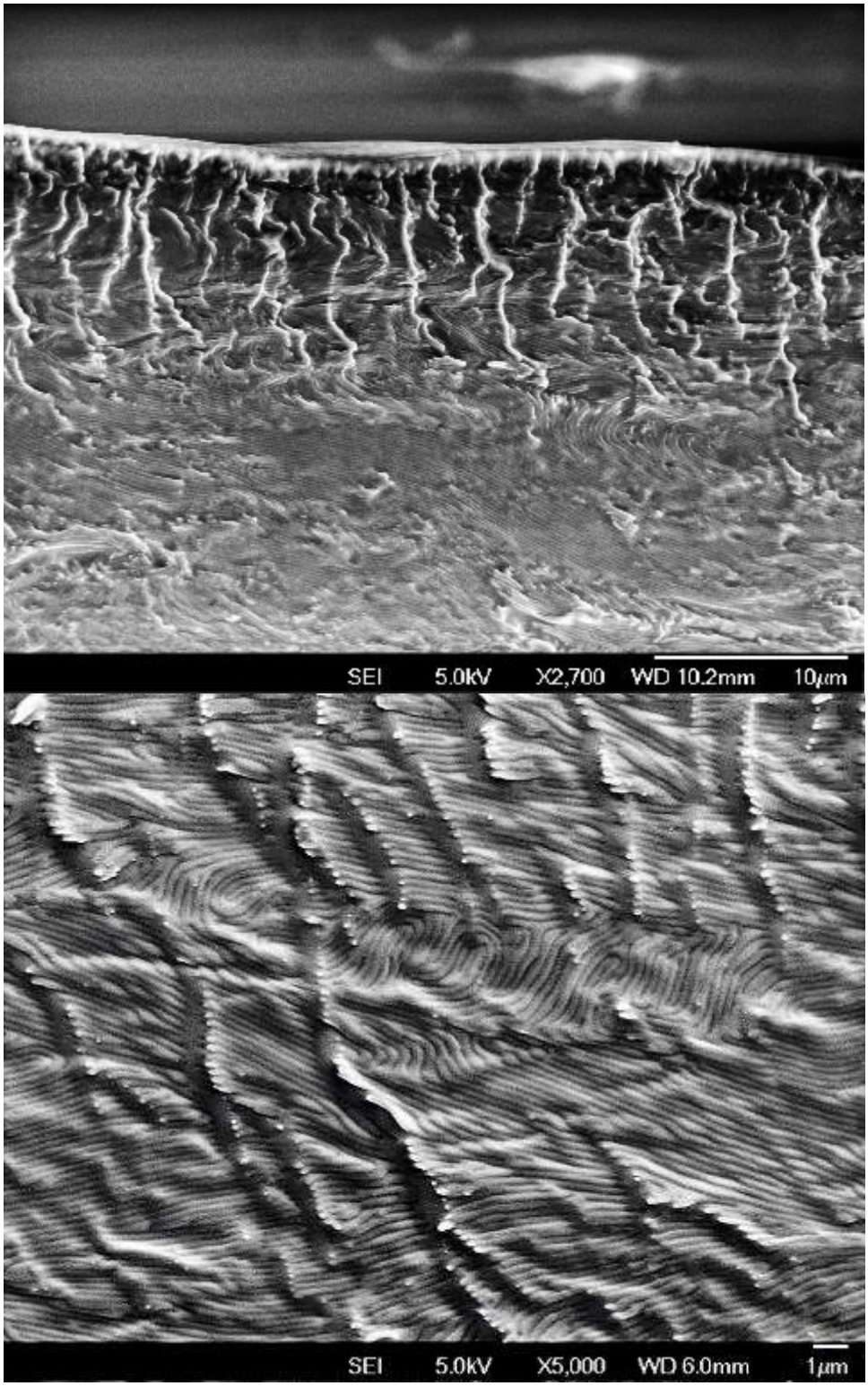


Figure S20. SEM of Dropcast Samples

Printed Samples – cross-sections of samples shown in **Figure 2C** of the main text

70°C – 30 mm/min – 30 kPa

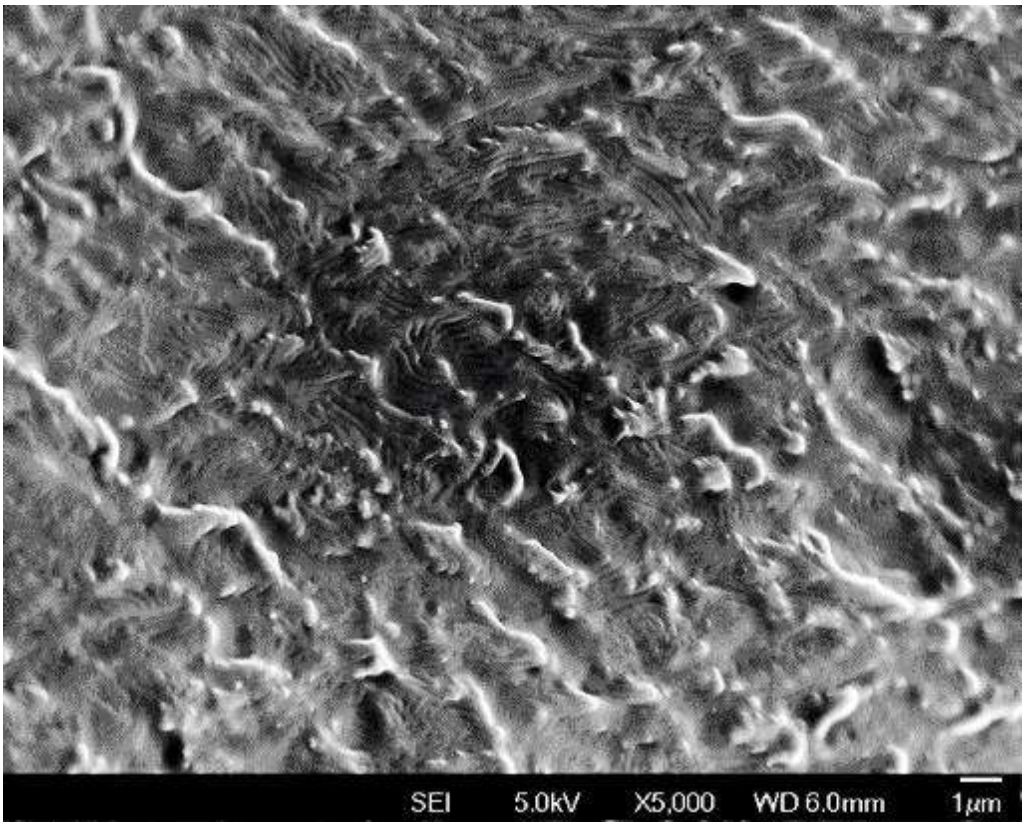


Figure S21. SEM of printed sample – 70°C – 30 mm/min – 30 kPa

50°C – 30 mm/min – 30 kPa

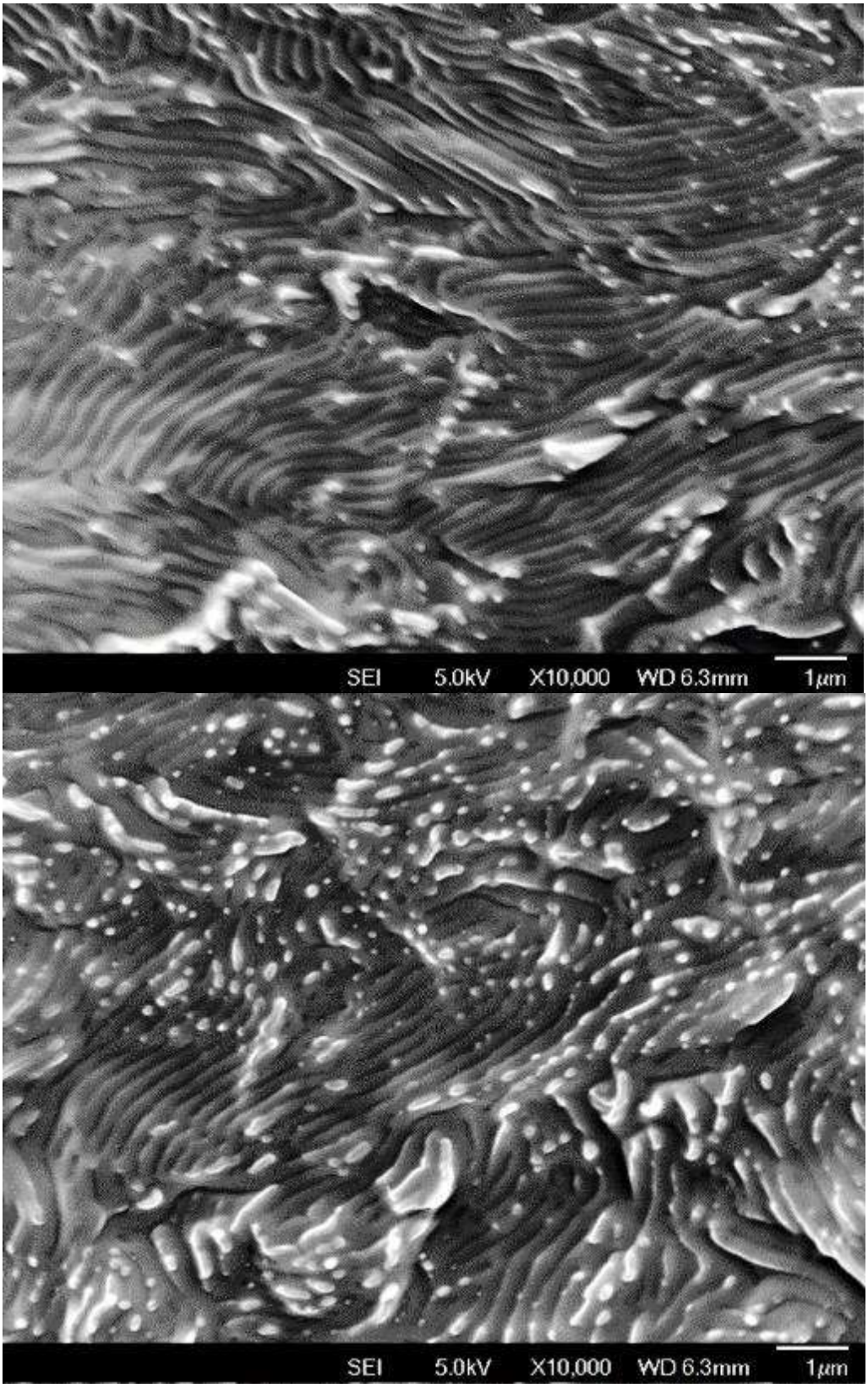


Figure S22. SEM of printed sample – 50°C – 30 mm/min – 30 kPa

25°C – 60 mm/min – 30 kPa

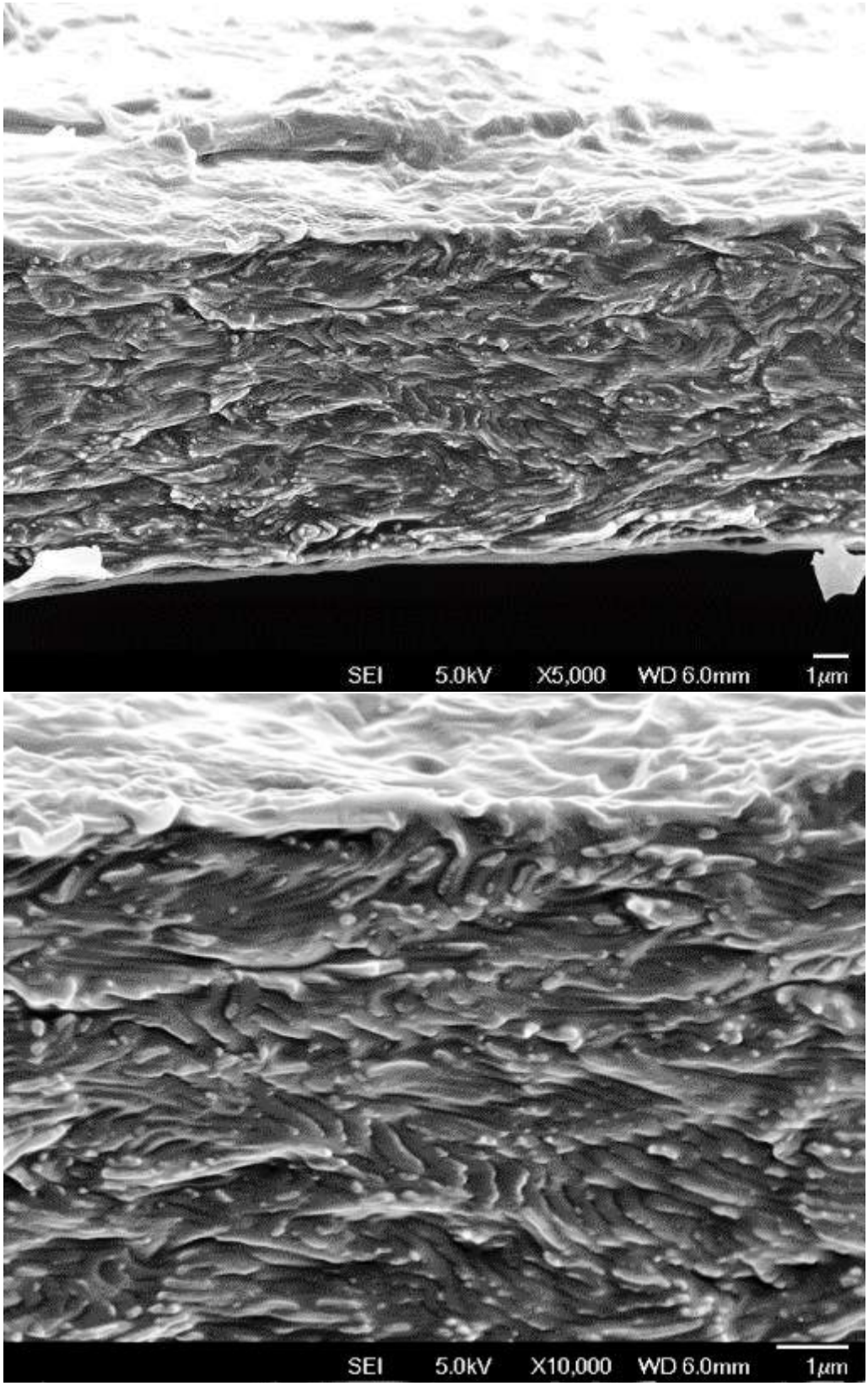


Figure S23. SEM of printed sample – 25°C – 60 mm/min – 30 kPa

§11. Small-Angle X-ray Scattering Analysis of Printed Films

Printed Films

Due to their high surface roughness/curvature, samples were unsuitable for running in grazing incidence configuration which is more typical for thin films. Instead, small-angle X-ray scattering experiments were performed in transmission mode at a shallow negative angle (**Figure S23)**. Experiments were performed at beamline 12-ID-B of the Advanced Photon Source (Lemont, IL) with beam energy 13.3 keV using the Pilatus 2M 2D detector. Five 0.1s scans were taken for each image shown below.

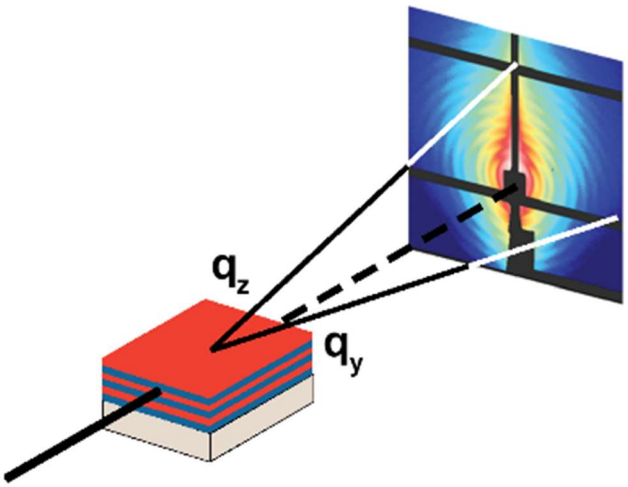


Figure S24. Schematic of printed film SAXS experiments.

2D Data Reduction

2D data reduction and scan averaging was performed using the Nika package developed by Jan Ilavsky[51] for Igor Pro (WaveMetrics, Lake Oswego, OR, USA). Spectra were restricted to the region of $q < \sim 0.4$ nm^{-1} before peaks were deconvoluted and fit using Igor Pro's built-in multi-peak fitting function. Peaks were fit sequentially from weakest to strongest as voigt functions before refitting the entire curve. The following figures correspond to points in **Figure 4C** of the main text and contain 2D data and fitted 1-D linecuts. Not all fitted peaks were used to calculate domain spacing.

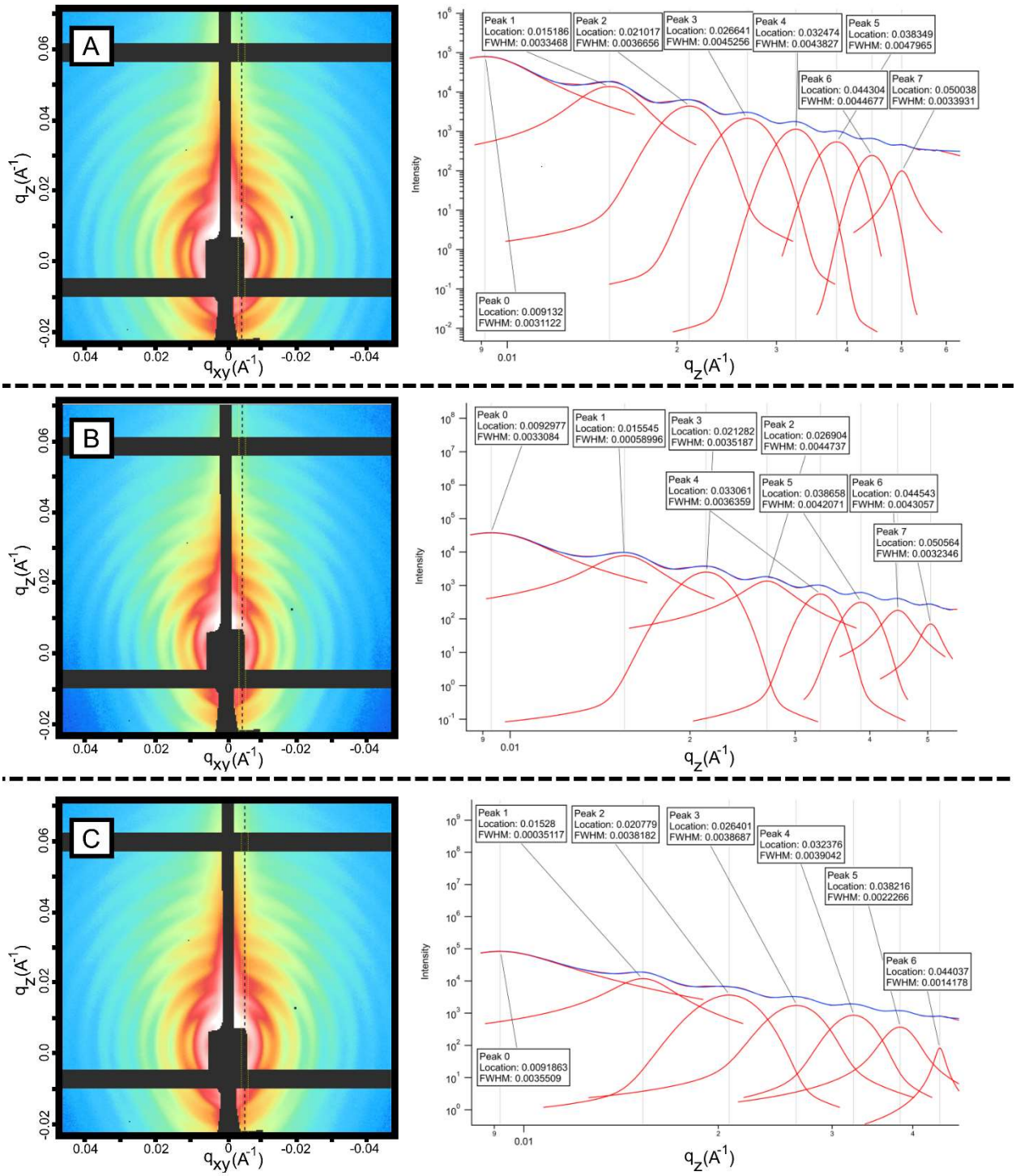


Figure S25. 2D SAXS data and fitted linecut for (A) Dropcast 1 (25C) position 1 (B) Dropcast 1 (25C) position 2 (C) Dropcast 1 (25C) position 3

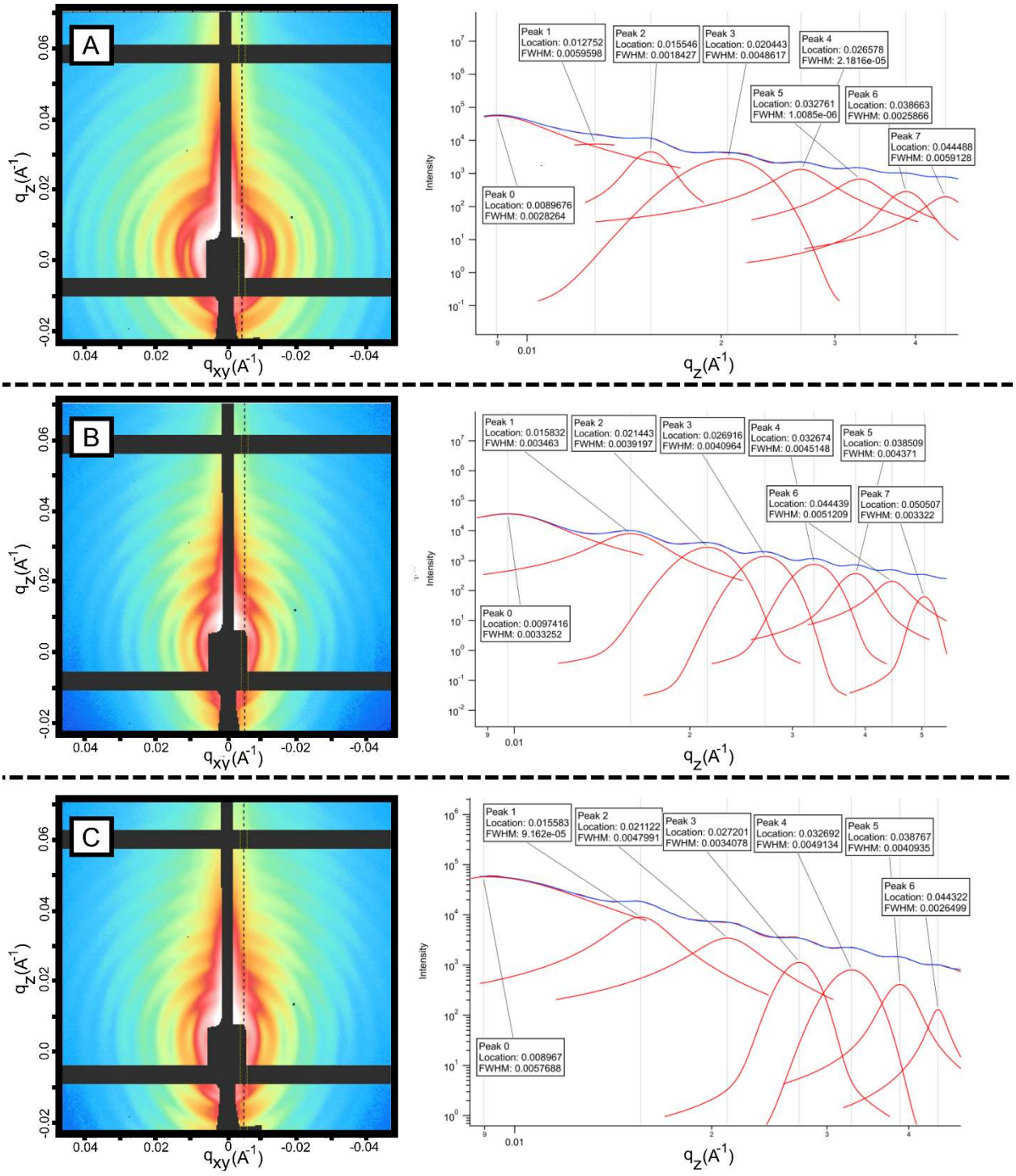


Figure S26. 2D SAXS data and fitted linecut for (A) Dropcast 2 (25C) position 1 (B) Dropcast 2 (25C) position 2 (C) Dropcast 2 (25C) position 3

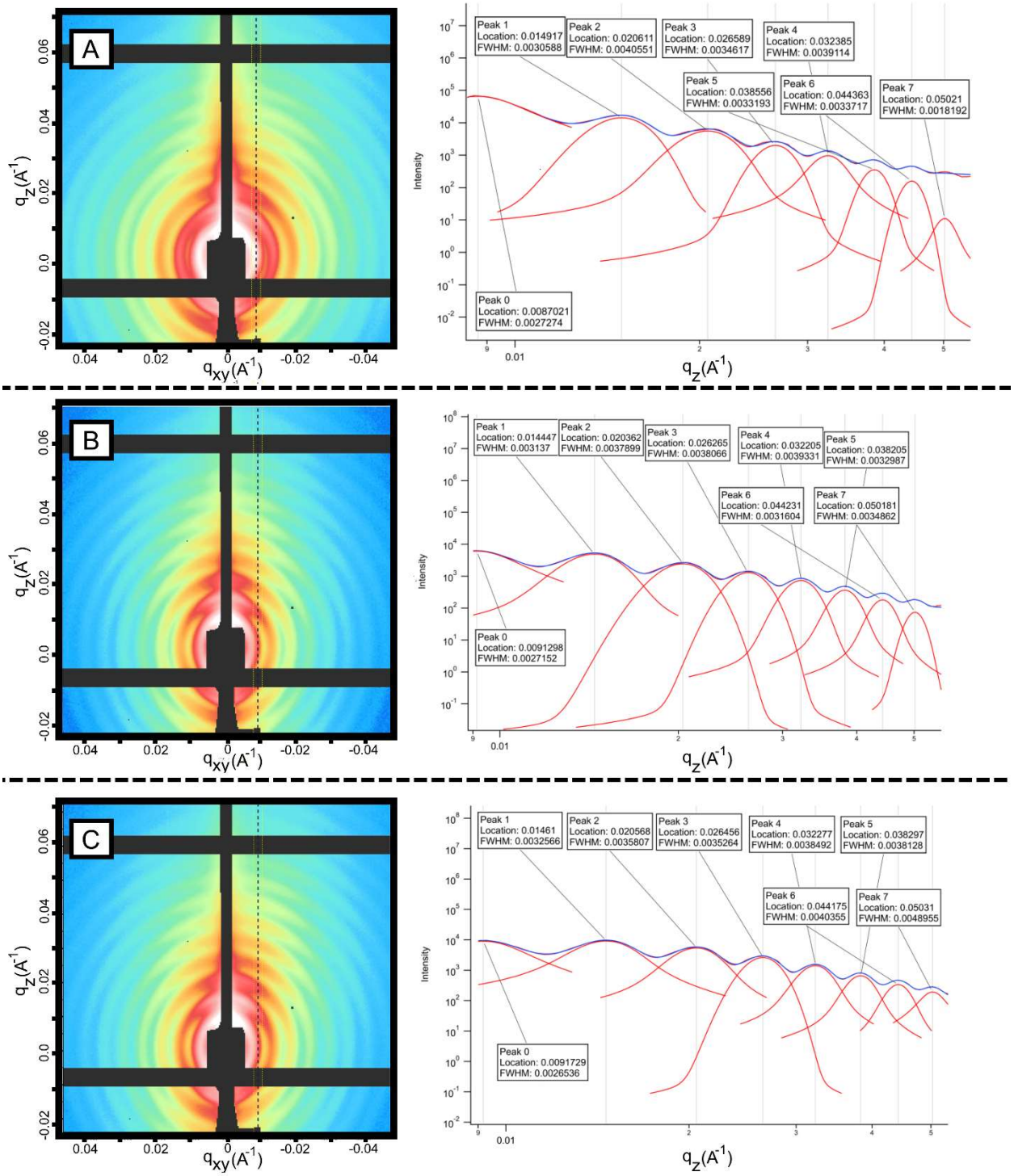


Figure S27. 2D SAXS data and fitted linecut for (A) Dropcast 3 (25C) position 1 (B) Dropcast 3 (25C) position 2 (C) Dropcast 3 (25C) position 3

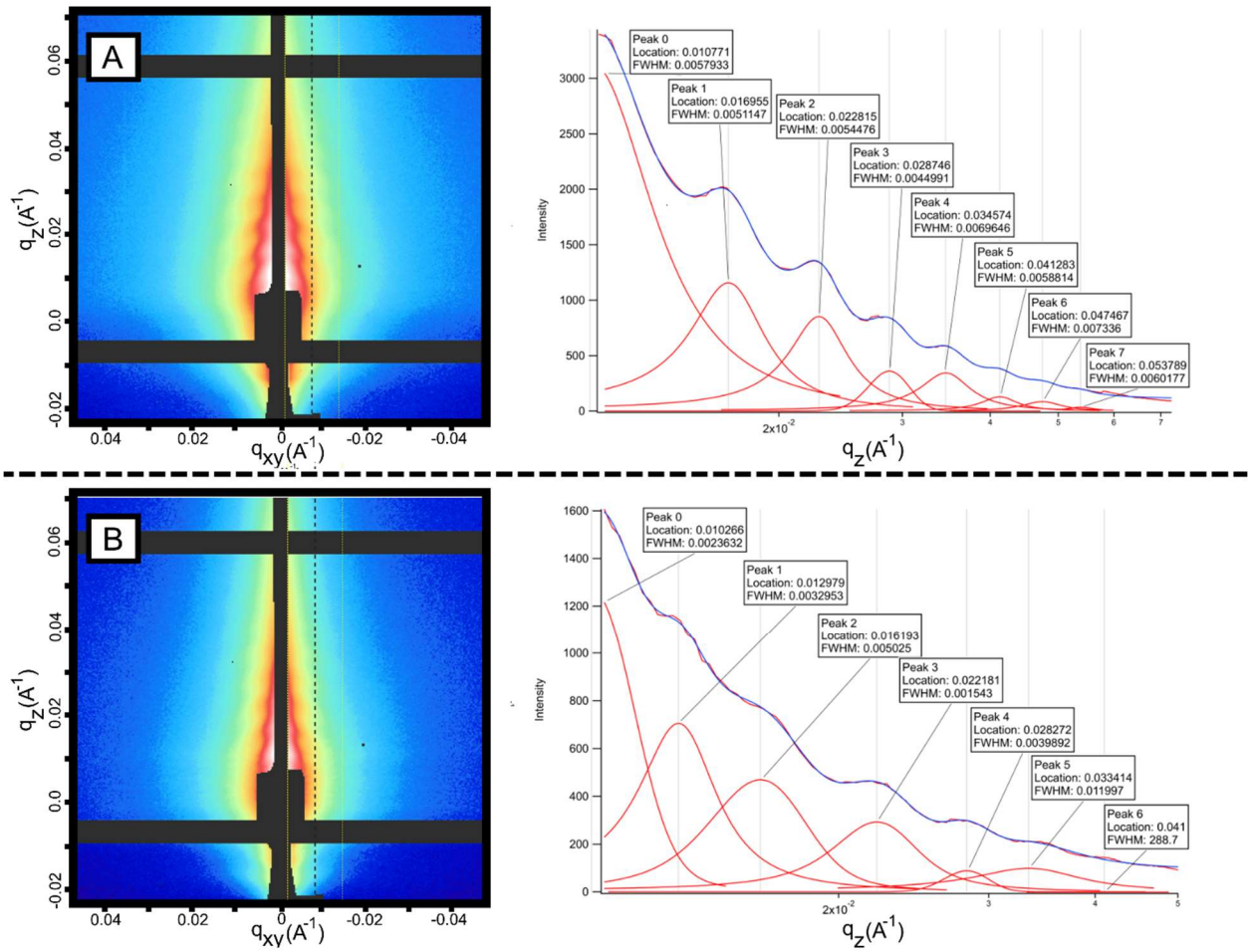


Figure S28. 2D SAXS data and fitted linecut for samples (A) printed at 15 mm/min, 25 C and (B) 15 mm/min, 25 C.

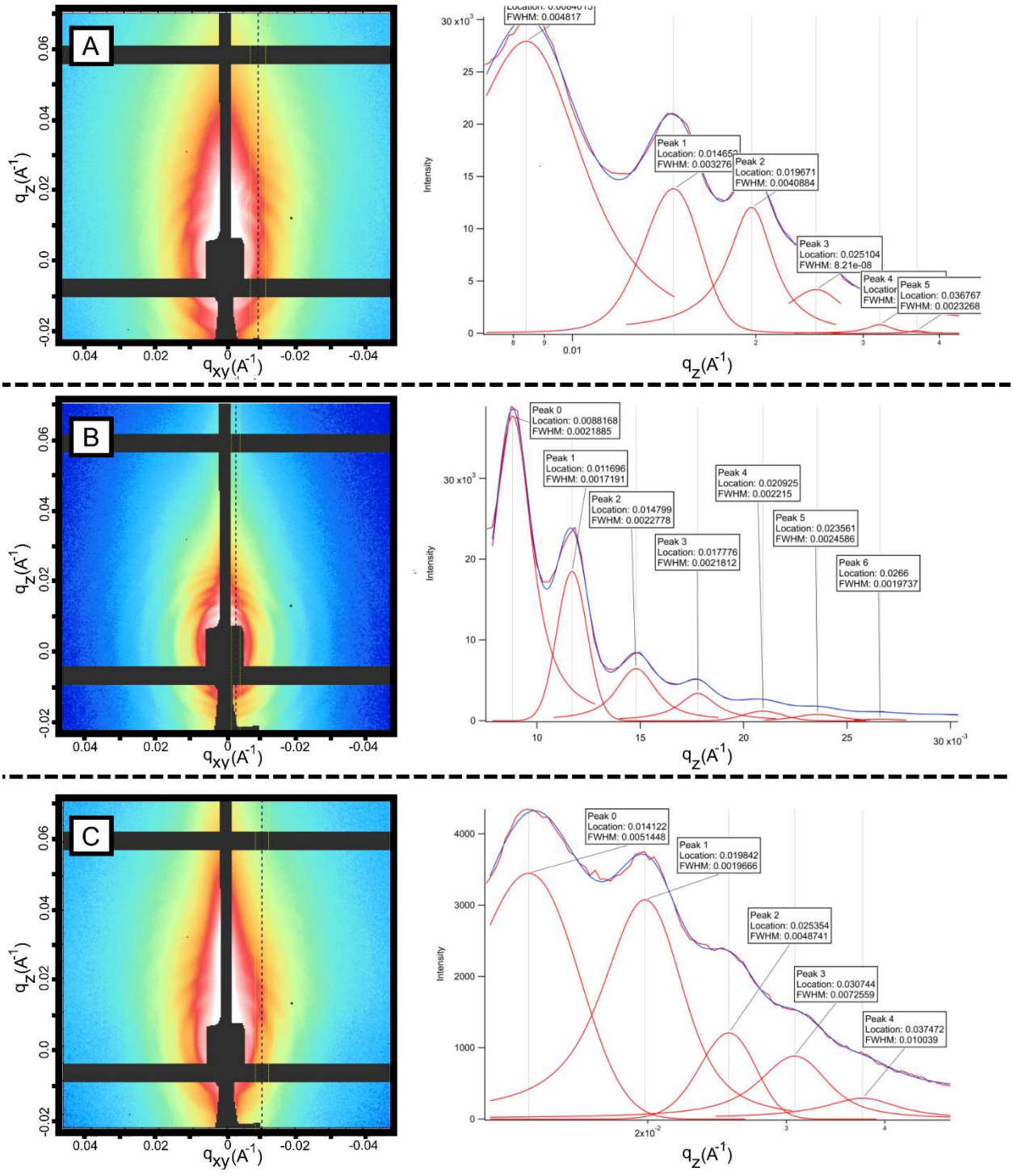


Figure S29. 2D SAXS data and fitted linecut for samples printed at (A)15 mm/min, 50 C; (B) 15 mm/min, 50 C; and (C) 30 mm/min , 50C

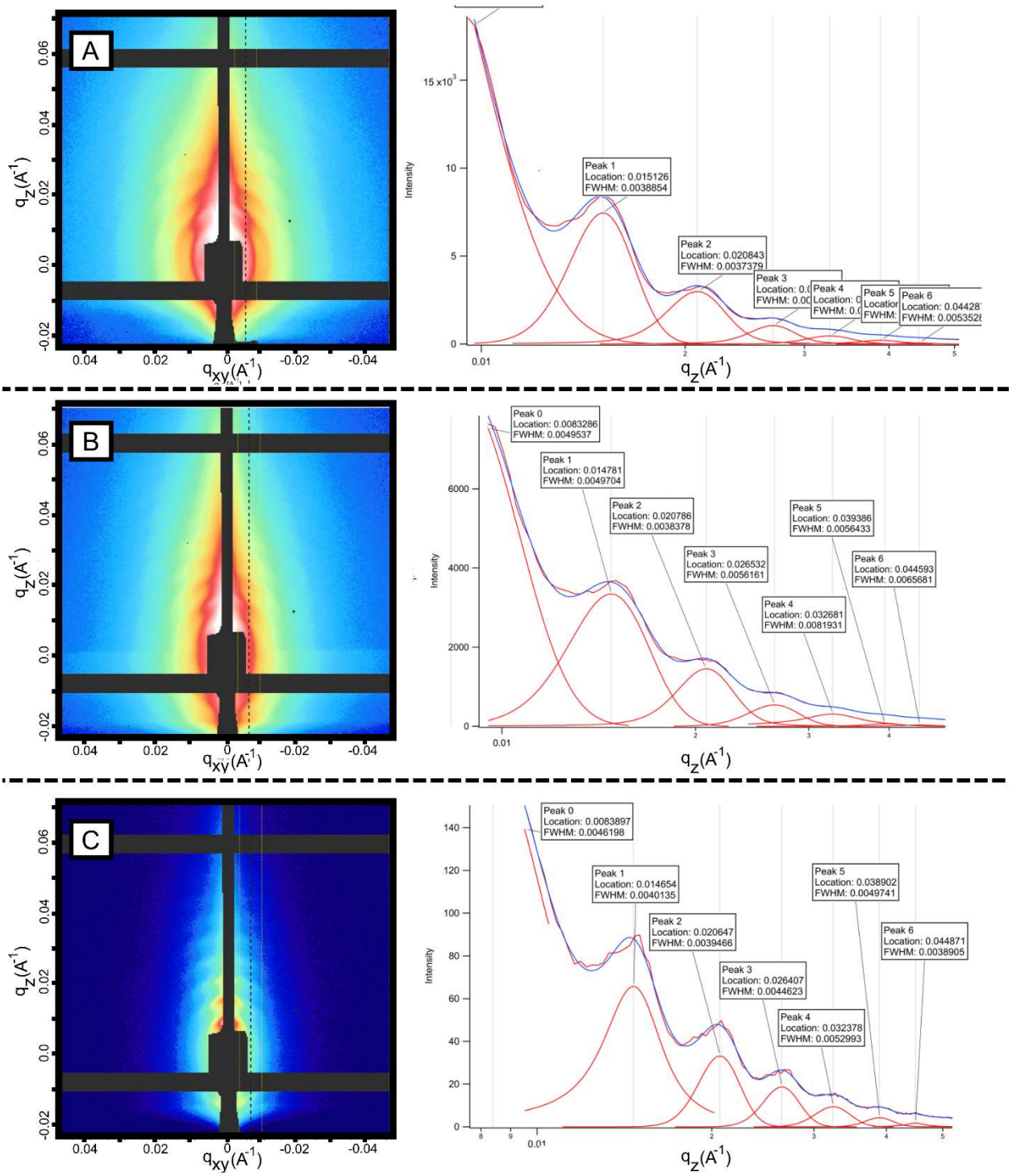


Figure S30. 2D SAXS data and fitted linecut for samples printed at (A) 60 mm/min, 50 C; (B) 60 mm/min, 50 C; and (C) 120 mm/min , 50C

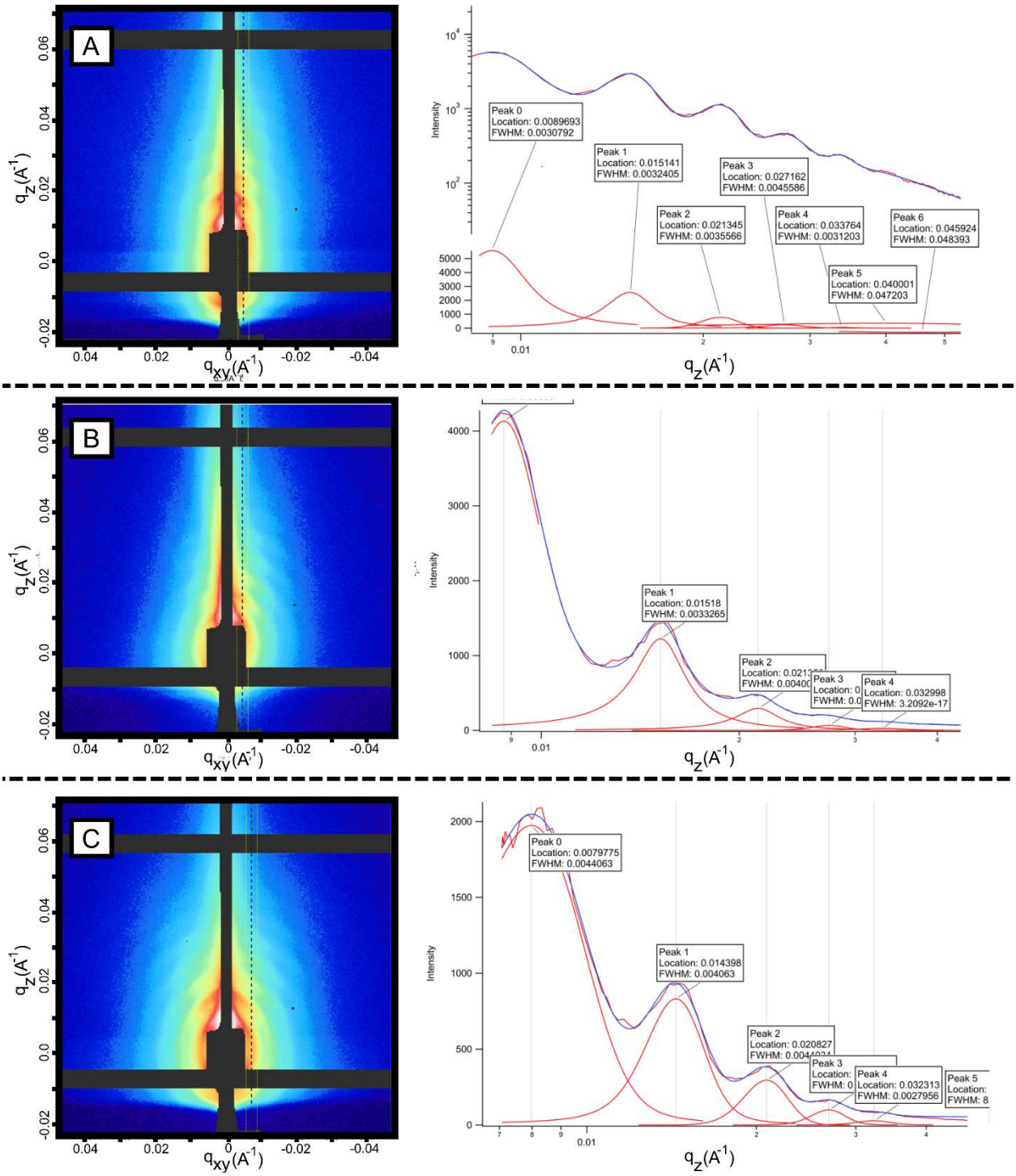


Figure S31. 2D SAXS data and fitted linecut for samples printed at (A) 180 mm/min, 50 C; (B) 240 mm/min, 50 C; and (C) 360 mm/min , 50C

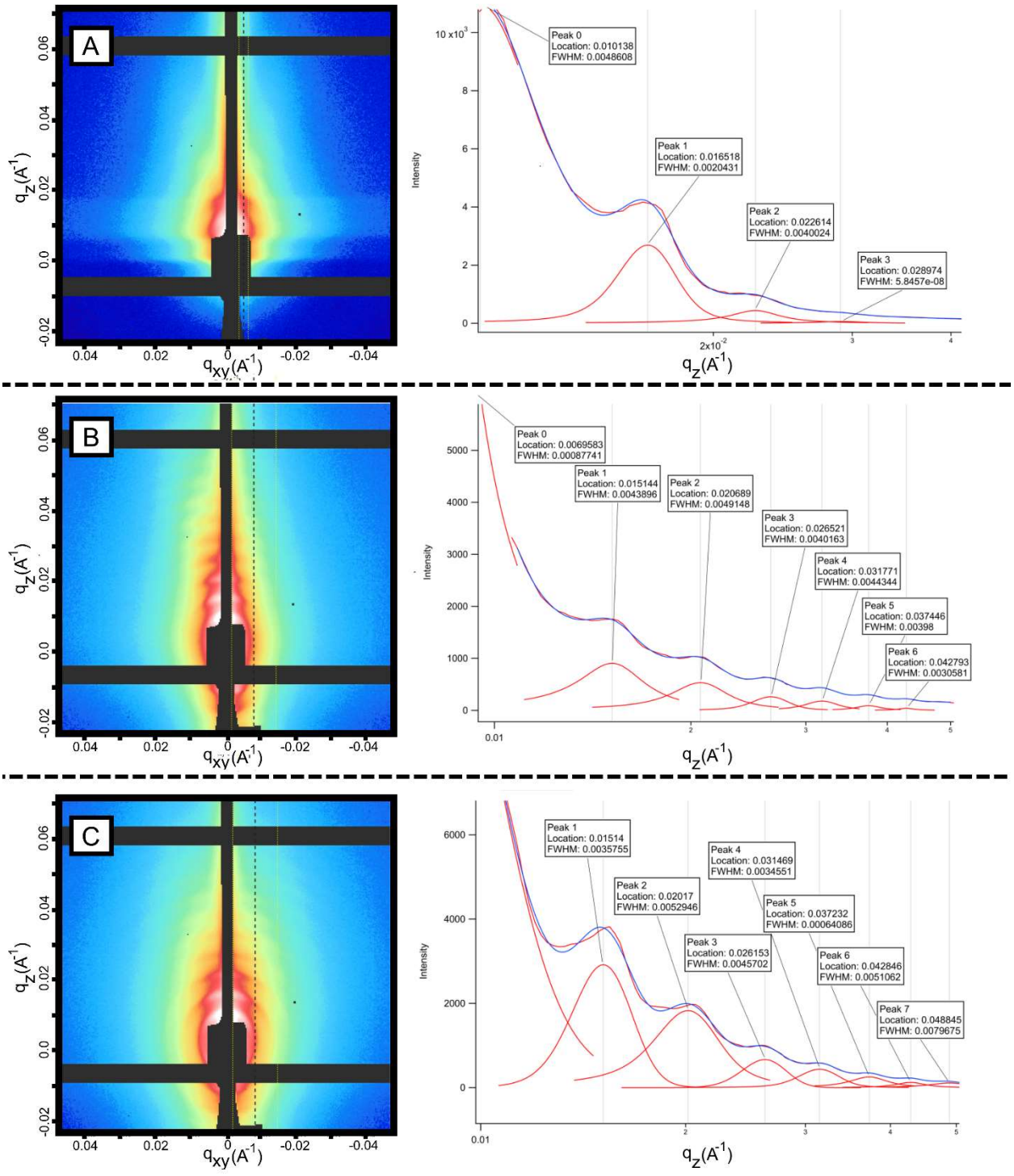


Figure S32. 2D SAXS data and fitted linecut for samples printed at (A) 600 mm/min, 50 C; (B) 15 mm/min, 70 C; and (C) 30 mm/min , 70 C

Determination of the 2D out-of-plane orientation parameter

To quantify the degree of out-of-plane alignment of lamellar domains, we have calculated the 2D out-of-plane orientation parameter[37] S_{2D} using the equation:

$$S_{2D} = < cos2\chi > = 2 < \cos^2 \chi > -1$$

Where χ represents the angle between the lamellar normal and the substrate normal. S_{2D} varies between -1 and 1, with 1 representing lamellae perpendicular to the substrate, 0 representing isotropy, and -1 representing fully parallel lamellae.

$< \cos^2 \chi >$ can be calculated from 2D scattering data by taking sector cuts and determining the area of a particular peak (here we choose n=2) as a function of χ and computing the following:

$$< \cos^2 \gamma >_{GIWAXS} = \frac{\sum I(\chi)_i \cos^2(\chi)_i \sin(\chi)_i \Delta\chi_i}{\sum I(\chi)_i \sin(\chi)_i \Delta\chi_i}$$

Where $\Delta\chi_i$ is the width of the ith sector.

We have performed this calculation for two samples: the dropcast film and the film printed at 15 mm/min at 50 C and calculate 2D orientation parameters of -0.73 and -0.96, respectively. This suggests that while both cases have lamellae oriented primarily parallel to the substrate, the printed films are more strongly oriented. This supports the assessment we have made in the manuscript and suggests that domain orientation plays a more minor role compared to domain size.

Domain Spacing Analysis

For lamellar morphology, constructive interference from successive layers causes peak spacing to follow

$$q_n = \frac{2\pi}{d_x} \qquad \text{Equation S2}$$

Where n is the order of the diffraction peak and d_x is the lamellar repeat distance. When the lowest order of diffraction is clearly visible, n takes the value 1, and the equation can be solved directly for lamellar repeat distance. In our case, the predicted domain size (~230 nm) puts the estimated 1st order peak at q = 0.0027 A^{-1}, which is below the minimum q-value we can detect. By rearranging equation 2 in series form, however, an alternate approach can be taken based on the difference in peak location between successive order peaks.

$$d_x = \frac{2\pi}{q_{n+1} - q_n} \qquad \text{Equation S3}$$

Here, we present domain sizes obtained by averaging the predicted spacing from at least 4 diffraction orders to minimize the impact of diffuse scattering in biasing the low q peak positions. For lamellar domains of a real symmetric block copolymers, peaks appear at integer multiples of the fundamental peak, with every other peak (n=2, n=4, n=6) significantly weakened[52]. In our case we predict that ϕ_{PDMS}~ 0.61 based on molecular weight, synthetic parameters, and the bulk density of the arm species (calculation shown above), and we find that for most of our spectra even order peaks are only visible weakly or as peak shoulders at low q. When even order peaks are weakly resolved, we instead fit only the stronger and highly evident odd numbered peaks and

multiply the resulting peak spacing by two to reflect the true domain spacing of the lamellar repeat unit.

We note that is **not** the approach commonly used for BBCP lamellar X-ray analysis, where 1st order peaks are usually experimentally measured[9,17]. We justify this approach in various additional ways below:

Justification 1: Arguments based on symmetry and SAXS theory

As mentioned above, this argument relies on the fact that for symmetric lamellae, SAXS spectra should reflect a nonmonotonic (alternating) decay of peak intensity with peak order. The underlying theory and calculations to support this can be found in many elementary SAXS texts, such as Roe[52]. We note that this qualitative behavior is strictly true only for symmetric lamellae: for asymmetric lamellae a monotonic decrease in peak intensity is predicted. As noted above, we calculate a PDMS volume fraction of $\phi_{PDMS} = 0.61$, which is nearly, but not perfectly symmetric. Aside from the inherent uncertainty in this value due to the choice of bulk density estimates, we have deliberately made several choices during measurement and analysis that further reduce the observed even order peak signal.

First, all samples manifest a very intense, diffuse scattering signal along q_z, requiring us to use a large vertical beamstop and obscure the completely vertically aligned data (where even-order peaks would be most evident). It is clear; however, that evenly spaced odd-order peaks are well-resolved far from the centerline, and furthermore data obtained at moderate q is less likely to be biased by diffuse scattering, film refraction, etc. As a result, we must choose a region of integration for peak fitting that is offset from the center line, where we can consistently fit the strong, well-resolved odd-order peaks.

For reference in the following discussion:
If 2nd order peaks are assumed to be hidden, $d^{h}_{X_{ray}} \sim 210\ nm$. Otherwise, $d_{X-ray} \sim 105\ nm$.

Justification 2: Arguments based on optical properties

The two indirect measurements of lamellar domain size that we can obtain from UV-Vis reflection measurements and cross-sectional scanning electron microscopy also support the assessment that even order peaks are obscured.

As described in the main text, we can use the ideal Bragg-Snell equation for 1D photonic crystals to estimate domain size from the peak reflected wavelength observed.

$$\lambda = 2(n_1 d_1 + n_2 d_2) \qquad \text{Equation S4}$$

Table S3. Parameters and source for substitution into the Bragg-Snell equation.

Parameter	Peak reflected wavelength (nm)	Refractive index of PDMS	Refractive index of PLA	Domain size of PDMS	Domain size of PLA
Symbol	λ	n_1	n_2	d_1	d_2
Value	Measured	1.40	1.46	To be solved	To be solved
Reference	-	33	34	-	-

We can relate the thickness of each layer based on our estimated volume fraction (ϕ) by assuming constant cross-sectional area.

$$\phi_1 = \frac{V_1}{V_1+V_2} = \frac{A*d_1}{(A*d_1)+(A*d_2)} = \frac{d_1}{d_1+d_2} \qquad \text{Equation S5}$$

Substituting in the value of $\phi_{PDMS} = 0.61$ and rearranging to solve for d_1 gives us:

$$d_1 = 1.57 * d_2 \qquad \text{Equation S6}$$

Now we can substitute this relation into our Bragg-Snell equation

$$\lambda = 2(n_1 * 1.57d_2 + n_2 * d_2) \qquad 35 \qquad \text{Equation S7}$$

and rearrange to solve for d_2

$$d_2 = \frac{\lambda}{2*(1.57*n_1+n_2)} \qquad \text{Equation S8}$$

Now we note that the X-ray domain spacing, d_T is equal to the sum of the thicknesses of each layer, that is:

$$d_T = d_1 + d_2 = 1.57\, d_2 + d_2 = 2.575 * d_2 \qquad \text{Equation S9}$$

By substituting in the relation for d_2 and our material properties we finally obtain:

$$d_T = \frac{2.57*\lambda}{2*(1.57*n_1+n_2)} = \ 0.351 * \lambda \qquad \text{Equation S10}$$

Our measured peak reflection of 403 nm – 626 nm, thus corresponds to estimates of d_T = 141.4 to 219.7 nm. As shown in **Figure 4c**, right of the main text (reproduced below), this agrees quite well with our X-ray values when hidden peaks are accounted for.

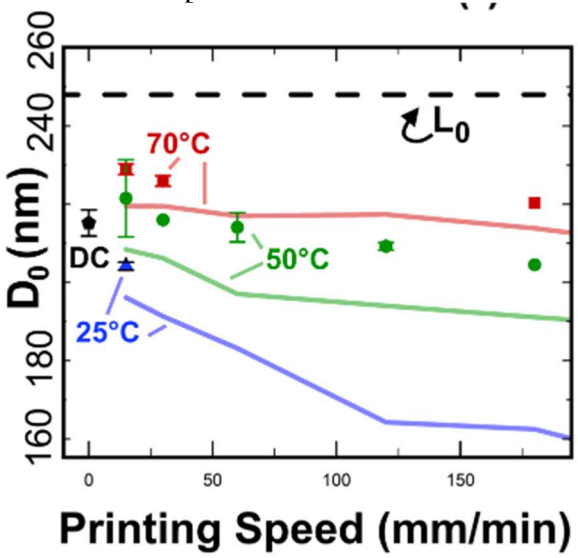


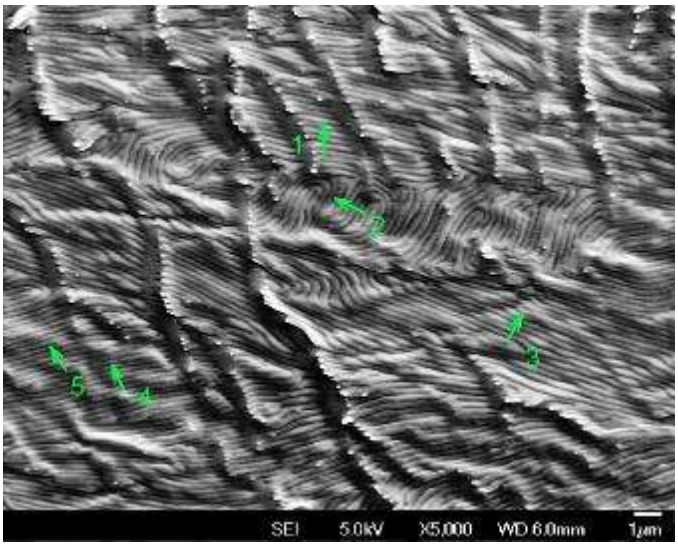
Figure S33. Calculated lamellar period versus printing speed. Where present, error bars on colored points represent the range of two sample scans. Error bars for dropcast (DC) sample represent the standard deviation of nine measurements across three samples. The dashed line represents the contour length of the bottlebrush estimated with a fixed backbone contour length of 0.62 nm per

norbornene repeat unit as per Dalsin et al.[9] Solid lines represent the domain size estimated from the optical data shown in **Figure 3** of the main text.

Justification 3: Arguments based on SEM-Cross sections

While we acknowledge the uncertainty of domain size measurements from cryo-fractured SEM cross-sections (detailed in the main text), it is useful to get a 'ballpark' value of domain size from the SEM as this technique is the most unambiguous way to determine domain size. The following analysis is performed using the free software ImageJ2[50].

Taking the image for which we have the best SEM contrast and most X-ray data (dropcast sample), we can directly measure the domain size (A+B layer repeat unit) as shown below. We first set the scale using the inset scale bar, and then use the measure tool.



MSMT	Length (nm)	Length/5 (nm)
1	1130	226
2	1161	232
3	973	195
4	1041	208
5	1077	215
Avg	1076	215
St. dev.	66	12

Figure S34: Annotated SEM of DC sample

The SEM measured domain size ($d_{SEM} = 215.27 \pm 12$ nm) compares quite favorably to our X-ray value accounting for hidden peaks ($d^{h}_{X_{ray}} \sim 210\ nm$).

Justification 4: Further Numerical analysis of X-ray data supports hidden peaks

Finally, it is possible to support the proposed method purely through numerical analysis of the X-ray data. Our strategy is as follows: first, we return to the equation for the location of a lamellar scattering peak in q-space.

$$q_n = \frac{2\pi n}{d_x} \qquad \text{Equation S11}$$

By rearrangement, we can identify the order of a measured peak based on the domain size calculated from the inter-peak spacing.

$$n = \frac{q_n * d_x}{2\pi} \qquad \text{Equation S12}$$

The calculated values of n should take on integer values (n = 1, 2, 3, …), however the calculated values of n for experimentally determined peaks always exhibit some remainder, which reflects

the accuracy of our peak fitting strategy. That is, an approach that leads to experimentally determined orders of n ~1,2,3,4 is more accurate than one that leads to orders of n ~1.5,2.5,3.5 etc.

Thus, by defining the cumulative error as the sum of the distances between the calculated order and its nearest integer, we can numerically compare the quality of the peak fitting approach when only the strong peaks are fit versus when the hidden peaks are considered. **Figure S35** plots the cumulative error of each approach for the first 5 peak orders of all of the samples printed at 50C.

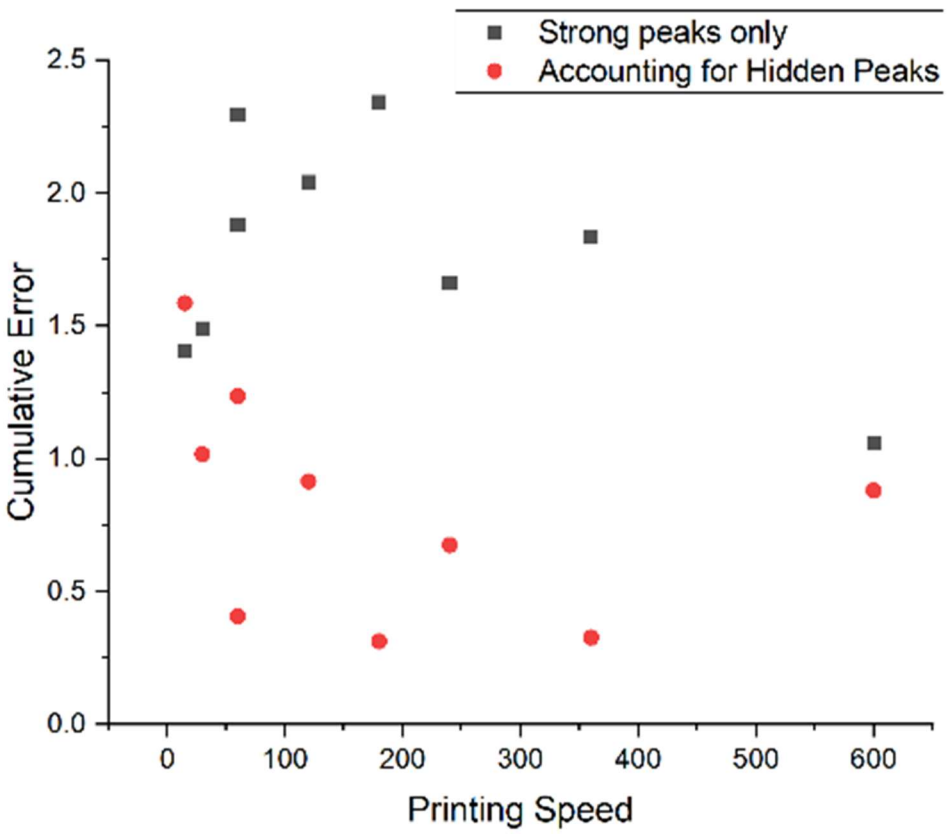


Figure S35. Cumulative error vs printing speed for X-ray samples printed at 50C with domain sizes calculated by each method.

It is apparent that by considering hidden peaks, we can drastically improve the quality of our fitted peak orders, providing further support for this decision.

§12. Topographical Mapping and Commensurability

Topography plots and selected correlations (to accompany the discussion in the main text)

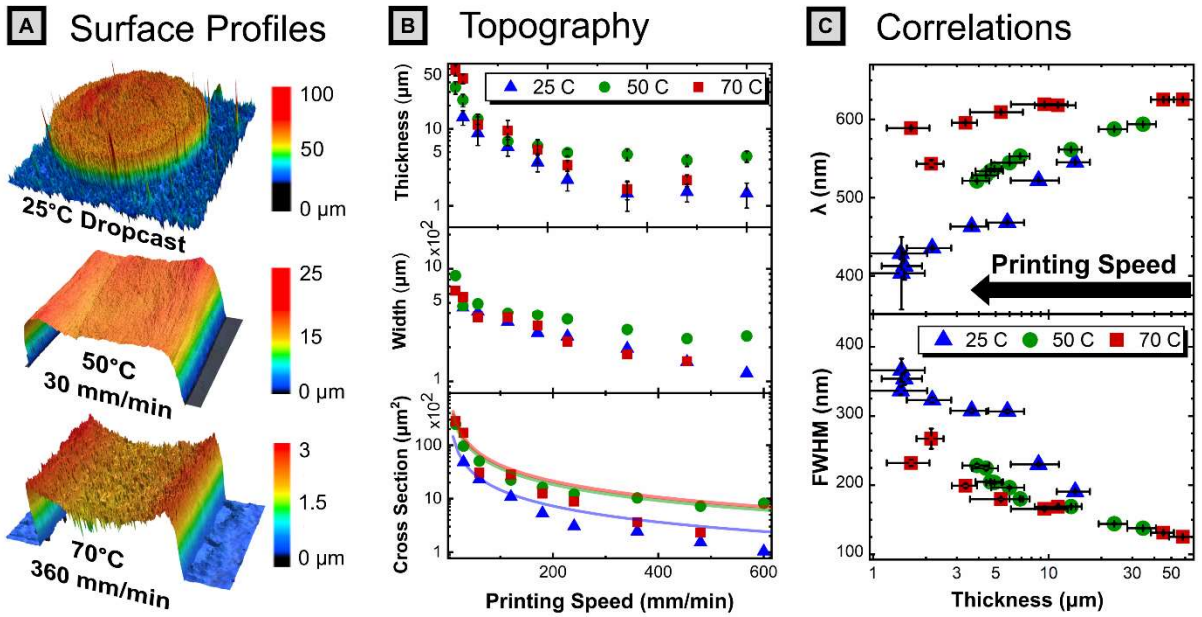


Figure S36. Topographical analysis of printed lines. (A) Surface profiles obtained using laser confocal optical profiling for samples printed at various speeds. (B) Compiled topographical data for each printing speed and temperature. Vertical error bars in the top panel reflect the rms roughness of the film. Solid lines on the bottom panel indicate expected scaling of cross section with printing speed for a Newtonian fluid. (C) Correlating the peak reflected wavelength and full-width half max to the film thickness. Horizontal error bars indicate RMS roughness. Vertical error bars indicate standard error of the optical peak fit.

Profile Data Collection Parameters

Optical profiling was performed using the Keyence VK-X1000 3D Laser Scanning Confocal microscope at the Illinois Materials Research laboratory. This technique was chosen after the sample was judged to be a poor fit for both stylus profilometry (Sloan instruments Dektak3ST) and atomic force microscopy (Asylum Research MFP-3D) due to its softness and waxiness/stickiness. Comparisons between film thicknesses obtained via optical profiling vs. stylus profilometry generally show only minor variation in average film thickness (< 5%). Film profiles were measured in laser confocal, "film top" mode with high brightness and a medium noise filter. Samples were scanned with substrate on both sides and plane leveled. RMS roughness values were evaluated using "stylus mode" with a tip radius of 5 microns.

Data analysis considerations (statistical analysis, spurious peak removal).

For film thickness and rms roughness measurements, a large ($\sim mm^2$) region was scanned. Within this region, 61 profiles were drawn perpendicular to the printing direction. For each profile, an average thickness and rms roughness are calculated. Then, the average of each of these 61 values per sample is reported as X-axis values and error bars, respectively, in **Figure S36**. The

full data set of film thickness, rms roughness, and standard deviations of each are reported in Table S4 and Figure S39 below.

We note that there was some initial difficulty scanning these semi-transparent samples due to the strong substrate reflection through the film. This manifests as spurious peaks [spikes] that are especially significant for thinner films. Scanning at sufficiently high magnification and slow speed helps reduce the number of spurious peaks.

We further mitigate errors caused by these peaks via three approaches:

1. Thresholding. The acquisition software can identify and remove data points that lay far away from the local mean value in each region. Because spurious peaks are very sharp and have thickness values near zero, this technique can very easily eliminate most spurious peaks. The software then interpolates thicknesses from surrounding pixels.
2. Manual removal of remaining low intensity peaks. These peaks have orders of magnitude lower intensity than surrounding regions are easily identified. Any remaining obvious spurious peaks (usually < 10 peaks) are removed manually and thickness in that region interpolated from surrounding pixels.
3. Averaging of many profiles. Spurious peaks generally occupy small, localized regions of the total scanned data. After steps 1 and 2, we then average 61 profiles spaced apart to cover the entire scanned region along the printed line). Even if some spurious peaks remain, their impact on the final thickness and rms roughness is thus very small.

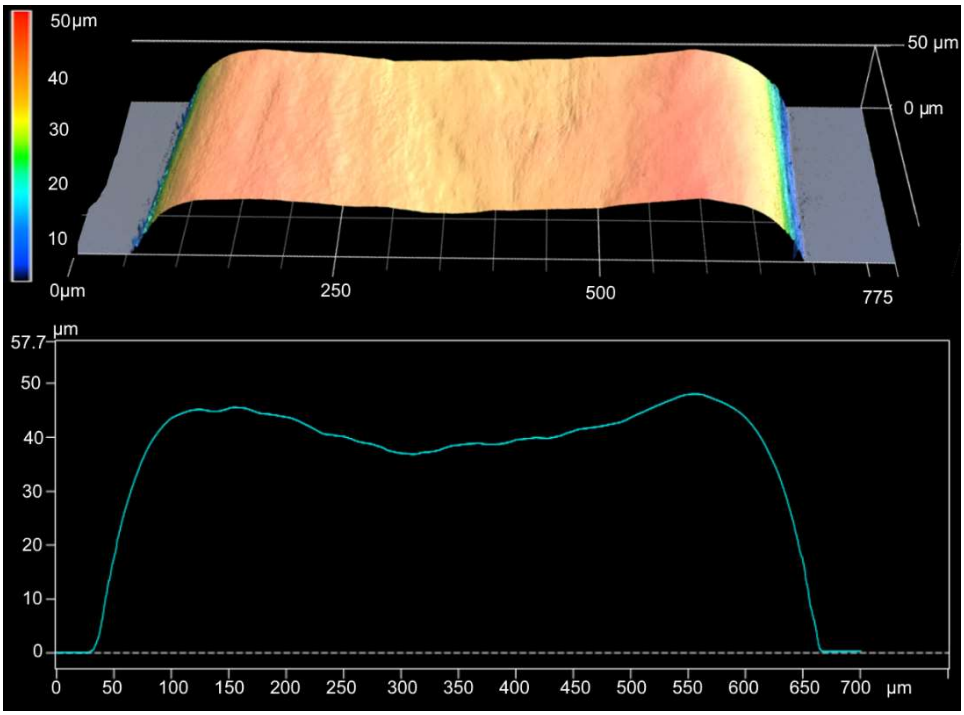


Figure S37. Sample optical profiling data.

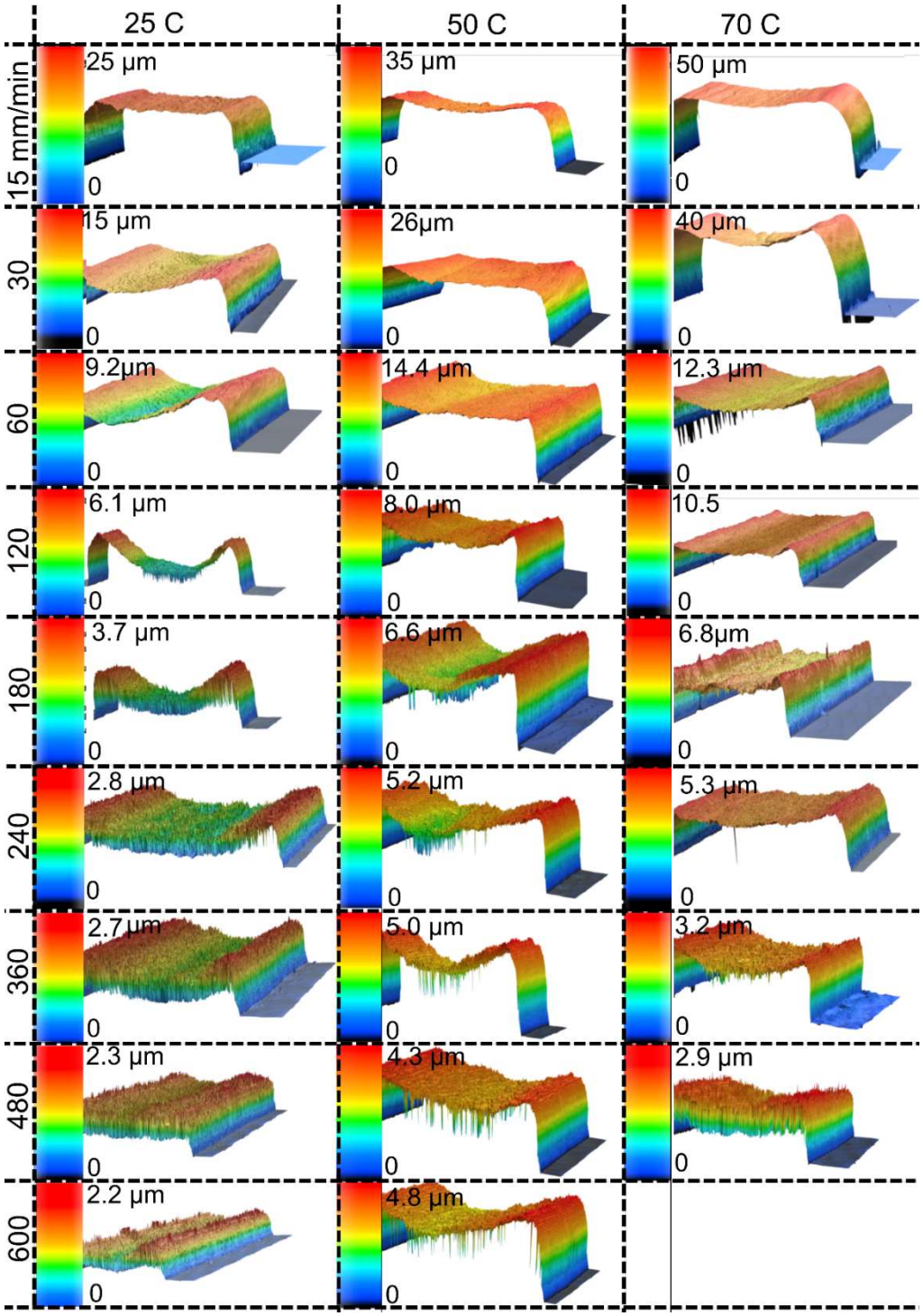


Figure S38. Raw optical profiling data for printed lines. Bed temperature is listed along the top (columns) and printing speed on the side (rows) Note: Horizontal and vertical axes are not at equal scale.

Table S4. Printed film thickness/roughness and standard deviations.

TEMP ($°C$)	PRINTING SPEED (mm/min)	THICKNESS (μm)	STD. DEV (μm)	RMS ROUGHNESS (μm)	STD. DEV. (μm)
25	15	23.74	1.02	4.91	0.47
25	30	14.17	0.58	3.03	0.35
25	60	8.76	0.13	2.67	0.05
25	120	5.83	0.18	1.41	0.04
25	180	3.65	0.09	0.89	0.02
25	240	2.17	0.06	0.62	0.03
25	360	1.44	0.20	0.59	0.03
25	480	1.51	0.08	0.39	0.04
25	600	1.45	0.10	0.52	0.03
70	15	58.17	0.34	9.57	0.23
70	30	45.04	5.99	6.73	0.37
70	60	11.30	0.36	2.96	0.04
70	120	9.47	0.17	3.38	0.05
70	180	5.35	0.21	1.79	0.07
70	240	3.36	0.69	0.55	0.05
70	360	1.65	0.29	0.45	0.02
70	480	2.15	0.12	0.38	0.03
50	15	34.52	0.34	6.41	0.12
50	30	23.67	1.02	4.36	0.58
50	60	13.43	0.27	2.00	0.11
50	120	6.89	0.69	0.90	0.10
50	180	5.99	0.07	1.28	0.07
50	240	4.91	0.12	0.50	0.04
50	360	4.67	0.04	0.83	0.03
50	480	3.91	0.04	0.68	0.02

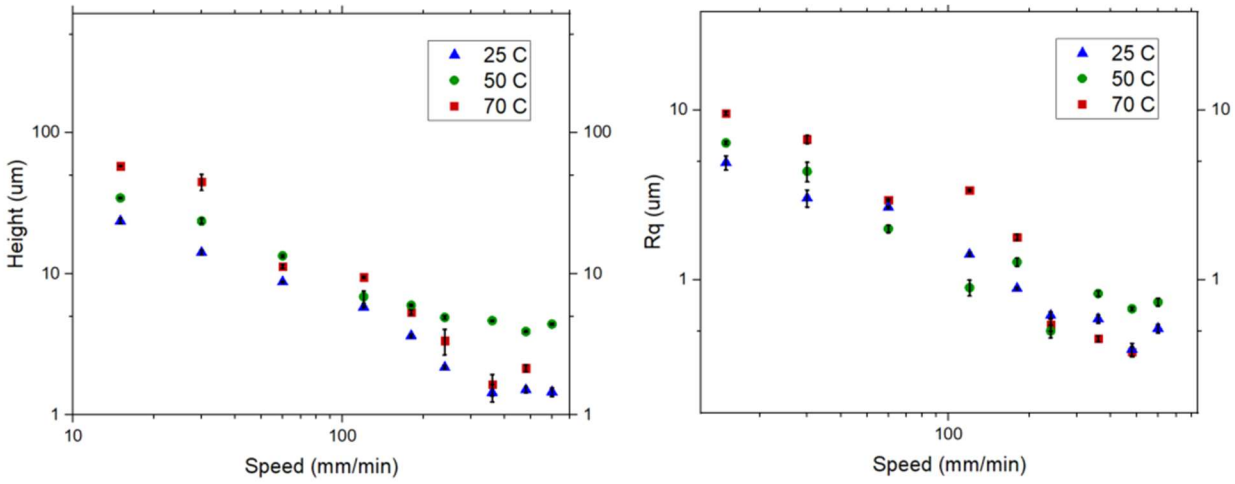


Figure S39. Plots of film thickness (left) and roughness (right) for printed meanderline patterns.

Discussion on Evaluating Commensurability arguments

To evaluate the importance of commensurability (balance of elastic and surface tension forces) in controlling domain d-spacing, we first plot domain size (estimated from optical measurements) versus film thickness.

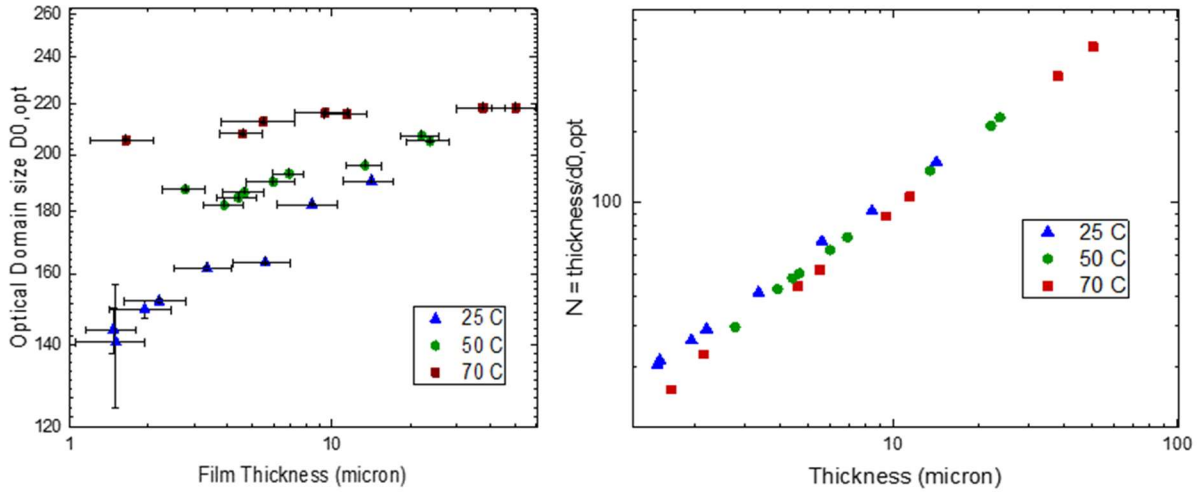


Figure S40. Left: Domain size estimated from optical measurements versus film thickness. Error bars denote film average roughness (X-error) and propagated error of optical peak fitting (Y-axis). Right: Number of PDMS-PLA layers versus film thickness.

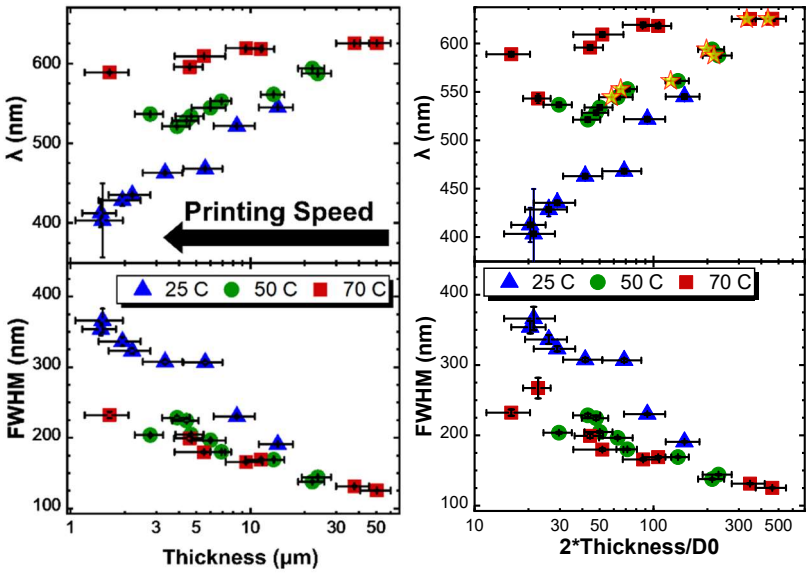
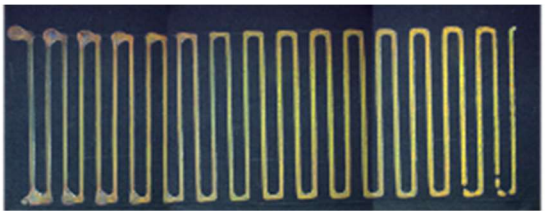


Figure S41. Measured wavelength plotted against both thickness (left panel, reproduced from **Figure S36**) and number of layers (2* thickness/D_0). In the right panel, triangles, circles, and squares represent division by $D_{0,opt}$ and stars represent division by $D_{0,X-ray}$. X-error bars represent error propagated from the uncertainty in thickness measurements and optical measurements.

§13. High Speed/High Pressure Printing Experiments

To gauge the impact of fluid flow driven assembly, experiments were performed in which both printing speed and pressure were increased. In this way, the wet film geometry could be kept constant, minimizing the impact of changing evaporation rates and any confinement effects. To determine the appropriate scaling between printing speed and pressure without performing in depth measurements to determine viscosity vs shear rate, a simple scaling experiment was performed. A long meanderline pattern was printed onto glass in which the printing speed was scaled by a factor of 1.41 for every pair of lines, and the pressure was scaled by a factor of 1.24. This scaling reflects the shear-thinning behavior of the fluid and was determined by trial and error to obtain the sample shown in **Figure S42**.

$V_f = 1841\ mm/min$
$P_f = 254.0\ kPa$

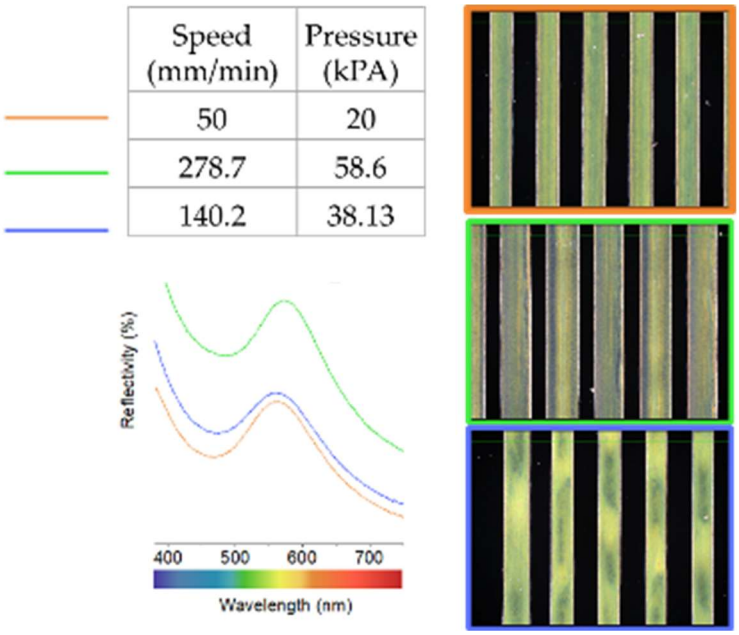


$V_0 = 15\ \frac{mm}{min}$
$P_0 = 12.5\ kPa$

Figure S42. Meanderline pattern with increasing pressure and printing speed.

Here, despite an extremeley large change in the printing speed and pressure, there is not a dramatic change in color until reaching very high pressures and speeds that were beyond the range used for the experiments in the main text.

To more quantitatively assess this phenomenon, three meanderline samples were printed at different speeds and pressure using this scaling relationship and their reflectivity was compared (**Figure S43)**.



Figures S43. Meanderline patterns printed onto silicon wafers. Despite a large increase in printing speed and pressure, there is very little change in the peak location observed.

§14. In Situ monitoring of printing and assembly kinetics

Discussion of the role of refractive index and domain size on reflected film color and intensity

By ascribing the structural color observed to lamellar domains, as observed in both SAXS and SEM, we can relate the measurable optical properties at normal incidence (Hue and Value) to the microstructural evolution through the equations for reflected wavelength (λ) and the reflected intensity (R) for a 1D lamellar stack (Equation S13,S14)[1].

$$\lambda = 2(n_1 d_1 + n_2 d_2) \qquad \text{Equation S13}$$

$$R = \left[\frac{1-\left(\frac{n_2}{n_1}\right)^{2N}}{1+\left(\frac{n_2}{n_1}\right)^{2N}}\right] \qquad \text{Equation S14}$$

Where n_1, n_2, d_1, d_2 represent the refractive indices of each layer and their corresponding thicknesses, and N represents the number of layers.

We note that these equations quantitatively describe only the observed reflectance from perfect 1D stacking under normally incident illumination and observation. As the film assembles/dries, this criterion is increasingly valid; however, in the initial stages it is likely that domains form at a variety of angles before reaching their final orientation, predominately parallel to the substrate as indicated by SAXS and SEM analysis. For dropcast samples where domains adopt a range of orientations, we note that at angles less than normal incidence, the apparent color of domains will be always be blue-shifted vs. that predicted by equation 1, but color evolution will follow the same trend during drying as the remainder of the film (initial red shift, followed by blue shift). A more accurate (quantitative) model of the color and intensity over time could be achieved by coupled finite-difference time-domain (FTDT), mass transport, and phase separation simulation which is out of the scope of this work. As a result, we use equations S13 and S14 above only for qualitative guidance and to not attempt to match results quantitatively.

During deposition, all three quantities (n, d, N) may vary, but by gauging their relative magnitudes it is possible to decouple their influence. For this system, the relevant refractive indices are $n_{PDMS} = n_{THF} = 1.40$ and $n_{PLA} = 1.46$. For solutions of polymer and solvent, the refractive index of the mixture lies between the two limiting refractive indices (of pure solvent and pure polymer), During the drying process, the refractive index of the PDMS layer remains unchanged (as it matches the solvent) and the refractive index of the PLA layer can change by only a maximum of ~4% as it goes from ~n_{THF} to n_{PLA}. By inspection of Equation 1, we can therefore note that the impact of refractive index change on wavelength during drying is a small redshift. We can then state that both the large red shift during the initial stages of drying, and the slight blue shift during the final stage are caused primarily by domain size variation. The increase in reflected intensity during drying is attributable to both an increase in the number of domains, and the influence of solvent removal on refractive index. As solvent dries, the refractive index of the PLA layer approaches its pure value (maximizing refractive index contrast) and the degree of microphase separation (number of domains) increase, with both phenomena possibly contributing to the observed saturation of intensity.

Methods

In situ optical microscopy was taken using an XIMEA xiQ USB3 camera connected to a high magnification Navitar lens system purchased from W. Nuhsbaum, Inc. Videos were obtained at moderate framerate (~45 fps) and analyzed frame-by-frame using a script written for MATLAB (The MathWorks, Inc., Natick, Massachusetts, United States). The following image shows the camera and lighting setup. All experiments in this series were carried out at room temperature with the same lighting/ fume extractor setup shown below.

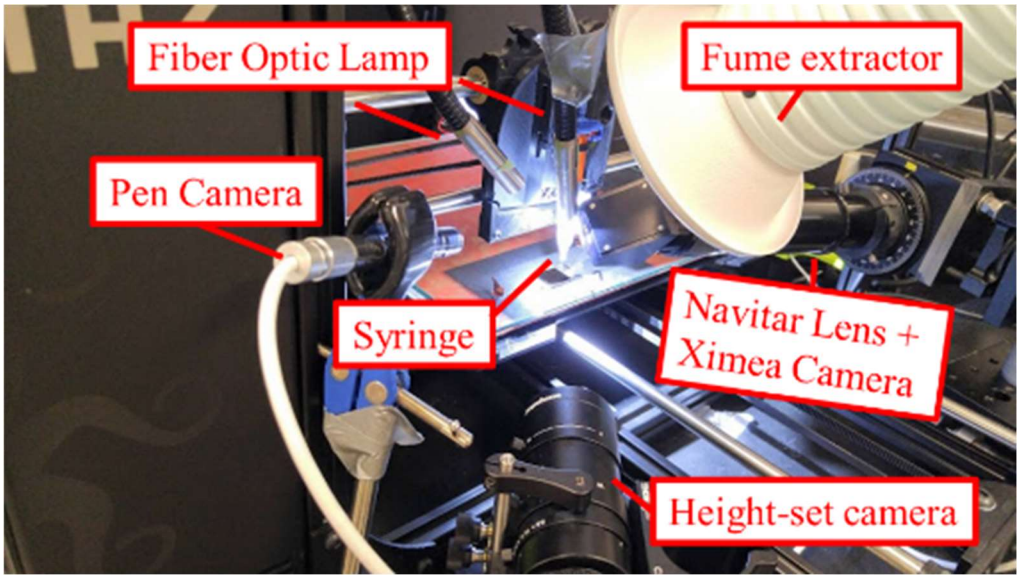


Figure S44. Experimental apparatus for *in situ* printing videos

We note that the conditions for these two sets of videos, while self-consistent, do not exactly match the printing conditions for those experiments from **Figure 2C** of the main text, as a high intensity light source was needed to ensure high framerate. This caused local heating and some added nonuniformity in the printed films; however, the films printed for these experiments show the same qualitative trend. Unfortunately, it was not possible to collect side view videos at sufficient resolution/framerate for analysis at speeds higher than 240 mm/ min. Photograph courtesy of Bijal Patel, University of Illinois.

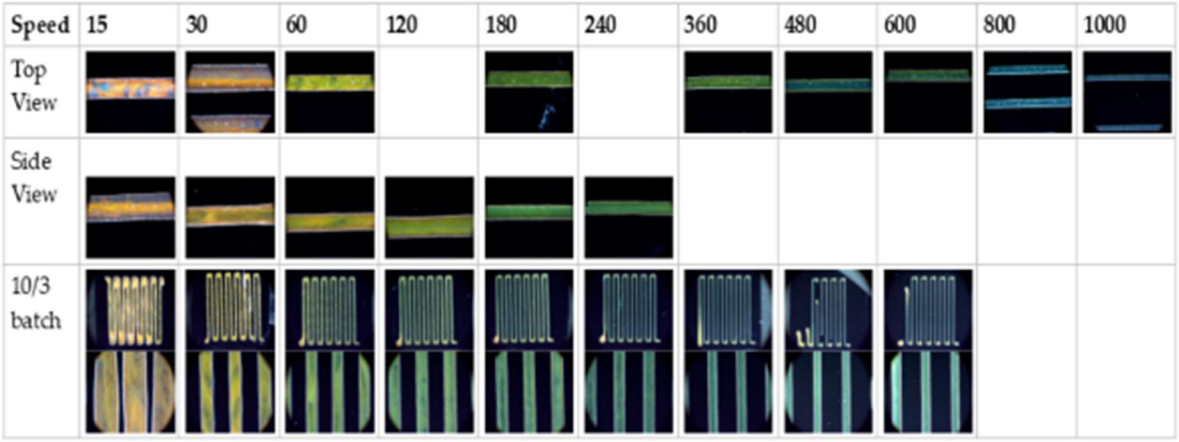


Figure S45. Microscope images of samples printed for different sets of experiments. "Top View" and "Side View" are for the corresponding kinetics videos under strong lighting, "10/3 batch" correspond to those samples used for peak quantification. Empty cells for top view 120,240 mm/min were samples that accidently disposed of before images could be taken.

Videos used for the following analysis are compiled in Supplemental Videos 5 and 6 as detailed at the beginning of the SI

Supplemental plot: Hue and intensity from top-view (reflection) printing videos.

Imaging difficulties prevent the use of Hue measurements as evidence for the mechanism. As the printed meniscus was very small, cameras and illumination had to be set up at extremely high brightness and away from normal incidence, using a different microscope camera than the drop-casting case (due to the printing nozzle). As a result, we find that the recorded video has a substantial red cast compared to the reflection (for example the final film shown below **Figure 5C, S45** actually appears green and not orange as in the video).

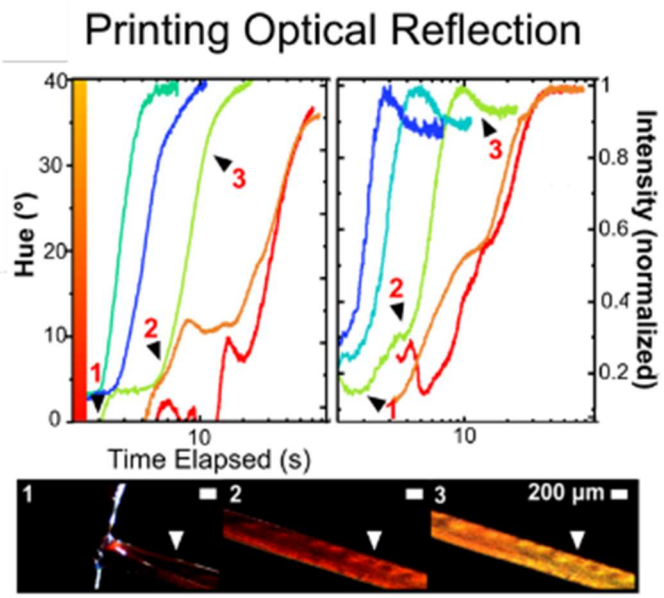


Figure S46. Calculated Hue and Intensity (normalized) curves from top-view (reflection) videos during printing. The limitations of the hue measurement are described in the main text.

Top View Analysis Strategy

Top view videos show the movement of the printing nozzle past a point on the substrate and track evolution of color of the printed line over time (**Supplemental Video 5).**

Videos were analyzed in MATLAB (analysis code follows). The analysis procedure is as follows:

- Step 0: Initialization of data directories.
- Step 1: Video file selected and export directories created
- Step 2: Video loaded into ram
- Step 3: Video converted to frames and exported.
- Step 4: The first frame for integration was chosen as the point at which the nozzle crosses the center of the field of view and an integration region (mask) is drawn.
- Step 5: Identify frame Tzero when needle enters mask for time correction
- Step 6: The program then iterates through all frames after time t=0, calculating the summed R,G,B, total, intensities within the masked region. If intensity exceeds threshold, value is set to NaN
- Step 7: Finally, the code exports this data and generates several useful plots (shown below).

Top View MATLAB Code

```
%Computes RGB/HSV Intensity from select region in videos

%% Step 0 - Initialize
    disp('Step 0: Initializing...')
    clear all
    close all

    % For rejecting oversaturated reflections from liquid
    % surface. BWThreshold is the ratio versus the average pixel intensity
    %within masked region, above which the pixel is omitted from analysis.
    %Determine by trial and error. White pixels should be completely removed.
    BWThreshold = 1.6;

    %Get current date/time
    dateTime = datestr(datetime(now,'ConvertFrom','datenum'));
    dateTime = strrep(dateTime,":","_");
    dateTime = strrep(dateTime,"-","_");
    dateTime = strrep(dateTime," ","_");

    %ENTER FOLDER PATH WITH ORIGINAL VIDEOS HERE
    filedir=strcat('C:\Users\...,'\');
       % or default to userdir
       % filedir=strcat(userpath,'\');

    newAnalysis = 0;

    disp('Choose Video File...')
    [filenameext, filedir]=uigetfile(strcat(filedir,'*.*'));
    [~,filename,fileext]=fileparts(strcat(filedir,filenameext));
    disp('Video File Chosen!')

    %Prompt for new analysis or not, if so, find log file
    inp = input("First Analysis? (Y/N): ",'s');
    if (strcmp(inp,'Y'))
        newAnalysis = 1;

        %Make specific folders for data/frame export
        expDir = strcat(filedir,'MATLABexport\',filename,"\");
        rawFramesDir = strcat(expDir,'frames\raw\');
        threshFramesDir = strcat(expDir,'frames\BrightThresholded\');
        calcDataDir = strcat(expDir,'calcData\',dateTime,'\');

        mkdir(expDir);
        mkdir(rawFramesDir);
        mkdir(threshFramesDir);
        mkdir(calcDataDir);


    else
```

```
        disp('Choose Log File')
        [logFileNameExt, logFiledir]=uigetfile(strcat(filedir,'*.*'));
        disp('Loading from Log File...')
        loadedData = readLog(strcat(logFiledir,logFileNameExt));
        loadParams = loadedData(:,1);
        loadedVals = loadedData(:,2);

        expDir = loadedVals(2);
        rawFramesDir = loadedVals(3);
        threshFramesDir = loadedVals(4);
        calcDataDir = strcat(expDir,'calcData\',dateTime,'\');
        framerate = str2num(loadedVals(6));
        maskFrame = str2num(loadedVals(7));
        disp('Data Loaded from Log File...')
    end

    calcDataDir = strcat(expDir,'calcData\',dateTime,'\');
    logParams = ["1. Export Directory" "2. RawFramesDirectory" "3.
ThresholdFramesDirectory" "4. CalculatedValuesDirectory"];
    logVals = [expDir rawFramesDir threshFramesDir calcDataDir];
    disp('Step 0: Initializing Complete!')

%% Load Video to RAM
    % If new analysis, will import video to RAM
    loadToRAM = input("Frame Export Required? (Y/N): ",'s');
    if (strcmp(loadToRAM,'Y'))
        disp('Importing Video to RAM...')
        obj = VideoReader(strcat(filedir,filenameext));
        vid = read(obj);
        frames = obj.NumberOfFrames;
        framerate = obj.FrameRate;

        logParams = [logParams "Framerate"];
        logVals = [logVals framerate];
        disp('Import Video Complete!')
    else
        disp('Skipping Loading Video to RAM')
        frames = size(dir(fullfile(rawFramesDir, '*.jpg')),1);
    end


%% Convert Video file to frames and export
    if(strcmp(loadToRAM,'Y'))
        disp('Export Raw Frames to File...')
        %Ask user if frames need to be output
        inp = input("     Output Frames? Y/N: ",'s');

        %if needs output, do so, else skip
        if (strcmp(inp,'Y'))
            for x = 1:1:frames

imwrite(vid(:,:,:,x),strcat(rawFramesDir,filename,'_raw_',num2str(x),'.jpg'))
;
                fprintf('\tExporting frame: %d out of %d.\n',x,frames);
            end
```

```
            disp('Export Raw Frames Complete!')
            beep
        else
            disp('Skipping Frame Export')
        end
    else
        disp('Skipping Frame Export')
    end

%% Draw mask
    disp('Masking')
    inp = input("     Draw New Mask? Y/N: ",'s');

    if(newAnalysis || strcmp(inp,'Y'))
        disp('Choose Frame to draw mask from.')
        %Choose frame to look at for drawing mask
        maskFrame =
chooseFrame(20,frames,rawFramesDir,strcat(filename,'_raw'),-1);

            %Export Frame
            expTail = strcat(filename,'_maskFrame.jpg');
            expPath = strcat(calcDataDir,expTail);
            saveas(gcf,expPath);

        disp("     Mask Frame chosen! Now Draw Mask");

        %Size of frame image

imsize=size(imread(strcat(rawFramesDir,filename,'_raw','_1','.jpg')));

        %Draw polygon mask
        %mx and my are the vertex positions of your mask
        %left click for points, right click to close loop
        %right click a vertex to create mask
        [mask, mx, my]=roipoly;

        %name image of frame to mask
        im
=imread(strcat(rawFramesDir,filename,'_raw_',num2str(maskFrame),'.jpg'));

        %Display mask overlayed onto image
        imshow(imoverlay(mat2gray(im),mask,[0.1 0.8 0.1]));

       %Save masked image
            %Export Frame
            expTail = strcat(filename,'_maskedFrame.jpg');
            expPath = strcat(calcDataDir,expTail);
            saveas(gcf,expPath);

        %Save mask itself
        %Save mask to file


            maskPath = strcat(expDir,'_',dateTime,'_mask.mat');
```

```
            save(strcat(calcDataDir,'mask_',dateTime,'.mat'),'mask');

        %uncomment for no mask
        %mask=logical(ones(imsize(1:2)));
    else %Load Mask from File
        disp('Choose Mask');
        [maskfilenameext, maskfiledir] = uigetfile(strcat(filedir,'*.*'));
        maskPath = strcat(maskfiledir,maskfilenameext);
        load(maskPath);
        save(strcat(calcDataDir,'mask_',dateTime,'.mat'),'mask');
    end

    logParams = [logParams "MaskFrame"];
    logVals = [logVals maskFrame];

    disp('Get Mask Complete!')


%% Identify first frame to calculate
    disp('Choose first frame to calculate values for (NOT tZERO)')

    %Choose frame for t=0
   calcStartFrame =
chooseFrame(maskFrame,frames,rawFramesDir,strcat(filename,'_raw'),mask);
        %Export start Frame
        expTail = strcat(filename,'_MaskedCalcStart.jpg');
        expPath = strcat(calcDataDir,expTail);
        saveas(gcf,expPath);

    disp('Frame for calculation start chosen!')

 %%Identify frameTzero time correction
    disp('Choose frame for time zero...')

    %Choose frame for t=0
    frameZero =
chooseFrame(maskFrame,frames,rawFramesDir,strcat(filename,'_raw'),mask);
        %Export Masked Frame
        expTail = strcat(filename,'_MaskedTzero.jpg');
        expPath = strcat(calcDataDir,expTail);
        saveas(gcf,expPath);

    disp('tzero Frame Chosen!')

%% Evaluate mean RGB intensity in masked region
    fstart=calcStartFrame;%number of starting frame
    fspace=1;%number of frames to increment by

    numFramestoCalc = frames-fstart; %number of frames to calc
    %Preallocate arrays
    Int_RGB = zeros(1,numFramestoCalc);
    frameAvg_Gint = zeros (1,numFramestoCalc);
    frameAvg_Rint = zeros (1,numFramestoCalc);
    frameAvg_Bint = zeros (1,numFramestoCalc);
```

```
    frameAvg_HSV = zeros (1,numFramestoCalc);

    %Folder for thresholded images
    threshfolder =
strcat(expDir,'frames\BrightThresholded\',num2str(BWThreshold),'\');
    mkdir(threshfolder);

    for k=fstart:fspace:frames

I_RGB=imread(strcat(rawFramesDir,filename,'_raw_',num2str(k),'.jpg'));

        %extract R, G, and B maps values separately
        R=double(I_RGB(:,:,1));
        G=double(I_RGB(:,:,2));
        B=double(I_RGB(:,:,3));

        %NaN values outside of region of interest
        R(mask==0)=nan;
        G(mask==0)=nan;
        B(mask==0)=nan;



        %Get avg RGB values for this frame within mask
        frameAvg_Rint(k-fstart+1)= nanmean(nanmean(R));
        frameAvg_Gint(k-fstart+1)= nanmean(nanmean(G));
        frameAvg_Bint(k-fstart+1)= nanmean(nanmean(B));

        %Mean intensity calculated from R,G,B.
        Int_RGB(k-fstart+1)=mean([frameAvg_Rint(k-fstart+1),frameAvg_Gint(k-
fstart+1),frameAvg_Bint(k-fstart+1)]);

        %NAN overexposed pixels (white) from light source reflection
        %White pixels are those above threshold value
        [lengthX, lengthY, rgb] = size(I_RGB);
        for column = 1:lengthY
           for row =1:lengthX
              if (sum([R(row,column) G(row,column) B(row,column)]) >
BWThreshold* 3*Int_RGB(k-fstart+1))
                  R(row,column)= nan;
                  G(row,column)= nan;
                  B(row,column)= nan;
              end
           end
        end

        %Export Thresholded Frame


imwrite(uint8(cat(3,R,G,B)),fullfile(threshfolder,strcat(num2str(k),".jpg")))
;
```

```
        fprintf('\tOn frame: %d out of %d processed.\n',k-fstart+1,frames-
fstart);
    end
    logParams = [logParams "Pixel Threshold"];
    logVals = [logVals BWThreshold];

    disp('Step 6: Calc Mean RGB Complete!')
%% Step 7 - Convert to HSV and Plot data
    disp('Step 7: Converting to HSV and Plotting Results...')

    %Time management

        %frame number array
        xFrame=fstart:fspace:frames;

        %time array
        %calc time with first frame at t = 0s
            xTime=(xFrame-frameZero)./framerate;%seconds
        %calc actual time of frame zero
            timeZero = frameZero/framerate;%seconds
        %correct xtime to account for timeshift
            xTime = xTime + timeZero;

        xvar=xTime;

    %Plotting mean RGB vs time
        subplot(2,3,1);
        plot(xvar,Int_RGB,'k')
        title('Mean RGB Intensity')
        xlabel('Time elapsed(s)')
        ylabel('Intensity (Arb)')
        set(gcf,'color','w');
        box on

    %Plotting R, G, B, individually and mean vs frame
        subplot(2,3,4);
        hold on
        plot(xFrame,Int_RGB,'k','Linewidth',1.5)
        plot(xFrame,frameAvg_Rint,'r','Linewidth',1.5)
        plot(xFrame,frameAvg_Gint,'g','Linewidth',1.5)
        plot(xFrame,frameAvg_Bint,'b','Linewidth',1.5)

        %add thick vertical lines every 10/100/1000 frames
        if (frames > 1000)
            for i=0:100:frames
                xline(i,'k');
            end

        elseif (frames > 10000)
            for i=0:1000:frames
                xline(i,'k');
            end
        else
            for i=0:10:frames
                xline(i,'k');
```

```
        end
    end

    title('RGB Separated Intensity Values')
    xlabel('Frame')
    ylabel('Intensity (Arb)')
    set(gcf,'color','w');
    hold off
    box on

%Plotting R, G, B, individually and mean vs time
    subplot(2,3,[2 3]);
    hold on
    plot(xvar,Int_RGB,'k','Linewidth',1.5)
    plot(xvar,frameAvg_Rint,'r','Linewidth',1.5)
    plot(xvar,frameAvg_Gint,'g','Linewidth',1.5)
    plot(xvar,frameAvg_Bint,'b','Linewidth',1.5)
    title('RGB Separated Intensity Values')
    xlabel('Time elapsed (s)')
    ylabel('Intensity (Arb)')
    set(gcf,'color','w');
    box on
    hold off
    set(gcf, 'Position',  [100, 100, 1000, 500])

%Convert RGB to HSV
    frameAvg_RGB = [frameAvg_Rint' frameAvg_Gint' frameAvg_Bint']./255;
    frameAvg_HSV = rgb2hsv(frameAvg_RGB);
    %Separate out hue / saturation/ Value
    frameAvg_H = frameAvg_HSV(:,1)*360;
    frameAvg_S = frameAvg_HSV(:,2);
    frameAvg_V = frameAvg_HSV(:,3);
    %"Unwrap" H circle so red values dont appear on both top and bottom
    %of axis
    for hValIndex = 1:size(frameAvg_H)
       if (frameAvg_H(hValIndex)>320)
         frameAvg_H(hValIndex) = frameAvg_H(hValIndex)-360;
       end
    end



    %Plotting H and V vs time
    subplot(2,3,[5 6]);
    hold on
    yyaxis left
    ylabel('Hue')
    plot(xvar,frameAvg_H,'k','Linewidth',1.5)
    yyaxis right
    plot(xvar,frameAvg_S,'b','Linewidth',1.5)
    plot(xvar,frameAvg_V,'r','Linewidth',1.5)
    title('Hue and Value over time')
    xlabel('Time elapsed (s)')
    ylabel('Value')
    set(gcf,'color','w');
    box on
```

```
        hold off
        set(gcf, 'Position',  [100, 100, 1000, 500])
        legend("Hue","Saturation","Value");

    %Concatenate results into matrix [frame xtime(s) Mean1(Arb) Mean2(Arb)
Rint(Arb) Gint(Arb) Bint(Arb)]
    m = [transpose(xFrame) transpose(xTime) transpose(Int_RGB) ...
        transpose(frameAvg_Rint) transpose(frameAvg_Gint) ...
        transpose(frameAvg_Bint) frameAvg_H frameAvg_S ...
        frameAvg_V];

    %Export Text
        expTail = strcat(filename,'_Data.txt');
        expPath = strcat(calcDataDir,expTail);

        %Write data file headers
        fid = fopen(expPath, 'w');
        fprintf(fid, "Frame,TimeElapsed(s),Mean Intensity,Red Intensity,Green
Intensity,Blue Intensity, Hue (degrees), Saturation (0-1), Value (0-1)\n");
        fclose(fid);

        %Write data
        dlmwrite(expPath,m,'-append','delimiter',',');

       disp('Step 7 Complete!')




%% Step 8:  Write Log File
   disp('Step 8 Writing Log File...!')

    %Write Log File

    if(writeLog(expDir,logParams, logVals))
        disp('Log File Written');
    else
        disp('Log Failed');
    end

%%

    %% END


disp('Sequence Complete!')


%% Local functions
%Writes parameters to log file
function logged = writeLog(expPath,params, vals)
    %Get current date/time
    dateTime = datestr(datetime(now,'ConvertFrom','datenum'));
```

```
    dateTime = strrep(dateTime,":","_");
    dateTime = strrep(dateTime,"-","_");
    dateTime = strrep(dateTime," ","_");

    logName = strcat(expPath,'\AnalysisLog_',dateTime,'_.txt');
    logFileID = fopen(logName,'w');
    fprintf(logFileID, 'Log Entry - Generated \n%s\n',dateTime);
    outputs = [params;vals];
    fprintf(logFileID, '%s\n',outputs);
    fclose(logFileID);
    logged = 1;
end

%Loads parameters from log file
function data = readLog(logFilePath)
    logFid   = fopen(logFilePath);
    txt = textscan(logFid,'%s','delimiter',',\n');
    fclose(logFid);
    txt = txt{1}; %Pull first element of cell aray
    txt = string(txt); %convert to string array
    %Convert to n x 2 array of param-val pairs
    data = [txt(1:2:end) txt(2:2:end)];
end


%Allows user to choose a frame
function pickedFrame = chooseFrame(frame, maxFrames, framesDir,
filename,mask)
    currentFrame = frame;
    pickedFrame = 0;
    done = 0;

    while (done ~= 1)
        %display masked, if mask given

        im
=imread(strcat(framesDir,filename,'_',num2str(currentFrame),'.jpg'));

        if (mask~=-1)
            imshow(imoverlay(mat2gray(im),mask,[0.0 0.5 0.0]));
        else
            imshow(im);
        end



        cmd = input('     q for done, asd+fgh for -(10/5/1)+(1/5/10), frames:
','s');
        switch cmd
            case 'q'
                done = 1;
                pickedFrame = currentFrame;
            case 'd'
                if (currentFrame > 1)
                    currentFrame = currentFrame-1;
```

```
            end
        case 's'
            if (currentFrame > 6)
                currentFrame = currentFrame-5;
            end
        case 'a'
            if (currentFrame > 101)
                currentFrame = currentFrame-100;
            end
        case 'f'
            if (currentFrame < maxFrames)
                currentFrame = currentFrame+1;
            end
        case 'g'
            if (currentFrame < maxFrames-5)
                currentFrame = currentFrame+5;
            end
        case 'h'
            if (currentFrame < maxFrames-100)
                currentFrame = currentFrame+100;
            end
        otherwise
            disp('Unrecognized input\n');
    end

end
end
```

Side View Analysis Strategy

Side view videos show the movement of the printing nozzle past a point on the substrate and track the meniscus height over time (**Supplemental Video 6).**

Videos were analyzed in MATLAB (analysis code follows). The analysis procedure is as follows:

- Step 0: Initialization of data directories.
- Step 1: Video file selected and export directories created
- Step 2: Video loaded into ram
- Step 3: Video converted to frames and exported.
- Step 4: User selects a frame and sets the Black/White threshold and noise filter strength to make the liquid film black and free space above it white. These can be set to be different for the early frames (usually these have different contrast to the following).
- Step 5: User sets the horizontal baseline manually.
- Step 6: User chooses what column should be used for height measurement (the highest black pixel in this column will be chosen as the meniscus height). User also selects the first frame for analysis.
- Step 7: User calibrates length based on a chosen standard in the field of view (i.e., width of the nozzle). This allows the code to report height values in real measurements.
- Step 8: Iterate through all frames and compute meniscus height at each location.
- Step 9. Compile data, export as plots and to file.

Side View MATLAB Code

```
%Meniscus Height profile from video calculator
% 12/19/2018 - JK
% 02/19/19 - BP

%% Step 0 - initialize
    disp('Step 0: Initializing...')
    clear all
    close all

     %ENTER FOLDER PATH WITH ORIGINAL VIDEOS HERE
     filedir=strcat('C:\Users…','\');
     %or default to userdir
     %filedir=strcat(userpath,'\');
     disp('Step 0 complete: Initialized gotten!')

    %Export directory
    expDir = strcat(filedir,'exportSide\');
    framesDir = strcat(expDir,'frames\');
    dataDir = strcat(expDir,'calcData\');
    disp('Step 0: Initializing Complete!')

%% Step 1 - Get Video file
    disp('Step 1: Get Video File...')
    [filenameext, filedir]=uigetfile(strcat(filedir,'*.*'));
    [~,filename,fileext]=fileparts(strcat(filedir,filenameext));
    disp('Step 1: Get Video File Complete!')

    %Make specific folder for data export
    dataExpDir = strcat(dataDir,filename,'\');
```

```
    mkdir(dataExpDir);

    %Make specific folder for frame export
    framesExpDir = strcat(framesDir,filename,'\');
    mkdir(framesExpDir);


%% Step 2 - Load Video to RAM
    disp('Step 2: Import Video to RAM...')
    obj = VideoReader(strcat(filedir,filenameext));
    vid = read(obj);
    frames = obj.NumberOfFrames;
    disp('Step 2: Import Video Complete!')

%% Step 3 - Convert Video file to frames and export
     disp('Step 3: Export Frames to File...')
     %Ask user if frames need to be output
     inp = input("     Output Frames? Y/N: ",'s');

     if(inp=='Y')
         for x = 1:1:frames

imwrite(vid(:,:,:,x),strcat(framesDir,filename,'\',filename,'_',num2str(x),'.
jpg'));
             fprintf('On frame: %d out of %d saved.\n',x,frames);

         end
     else
          disp('     Step 3 skipped!')
     end

imsize=size(imread(strcat(framesDir,filename,'\',filename,'_','1','.jpg')));
     beep
     disp('Step 3 complete: Frames Exported!')

%% Step 4 - SET BW Threshold and filtering
%Choose a frame then pick a BWthresh value. Iterate to find best value

    disp('Step 4 Set Black/White Threshold...')
    framenum= chooseFrame(20,frames,framesDir,filename,-1);

    %BWthresh is used to determine when converting from raw grayscale images
    %to black and white, so that we can identify the first black pixel in a
    %column as the top surface of the meniscus. Values range from 0 to 1. As
    %a first guess, the threshold is initialized at 0.4, but the user will
    %iteratively vary this until a clean image is obtained.
    BWthresh=0.4;

    %WF values reflect the strength of noise filtering applied to the image
    %Begin with a high value and reduce until the region above the meniscus
    %has no stray black pixels. The user will iteratively modify this at
    %runtime until a clean image is obtained.

    WF = 1;
    WF1=0;
```

```
I1test=imread(strcat(framesDir,filename,'\',filename,'_',num2str(framenum),'.
jpg'));
    Imid =
imread(strcat(framesDir,filename,'\',filename,'_',num2str(framenum+round(0.3*
(frames-framenum))),'.jpg'));
    Ifinal=imread(strcat(framesDir,filename,'\',filename,'_',num2str(frames-
5),'.jpg'));

    BWtest=rgb2gray(I1test);
    I2mid=rgb2gray(Imid);
    I2final=rgb2gray(Ifinal);

    montage({I1test,Imid,Ifinal,BWtest,I2mid,I2final}, 'Size', [2 3]);

    %let user enter BW and refresh image until they enter -1 for quit
    done = 0;
    while (done ~= 1)

        fprintf("Current Filter setting: BW %d, Filterx %d\n",BWthresh,WF);
        inp = input("Vary BWthresholdInit20/Final (A,B) or Filter (C/D):
",'s');


        switch(inp)
            case 'A'
                inp = input("Enter BWinitial threshold from 0 to 1 [no stray
black pixels], -1 to save: ",'s');
                inp = str2num(inp);
                BWthresh1 = inp;
            case 'B'
                inp = input("Enter BW threshold from 0 to 1 [no stray black
pixels], -1 to save: ",'s');
                inp = str2num(inp);
                BWthresh = inp;
            case 'C'
                inp = input("Enter number of times to run WFilter on 1st 20
frames: ",'s');
                inp = str2num(inp);
                WF1 = inp;
            case 'D'
                inp = input("Enter number of times to run WFilter on rest of
frames: ",'s');
                inp = str2num(inp);
                WF = inp;
            case 'q'
                done = 1;
            otherwise
        end

        %Display new setting

I1test=imread(strcat(framesDir,filename,'\',filename,'_',num2str(framenum),'.
jpg'));
        Imid =
imread(strcat(framesDir,filename,'\',filename,'_',num2str(framenum+round(0.3*
(frames-framenum))),'.jpg'));
```

```
Ifinal=imread(strcat(framesDir,filename,'\',filename,'_',num2str(frames-
5),'.jpg'));

        I2test=rgb2gray(I1test);
        I2mid=rgb2gray(Imid);
        I2final=rgb2gray(Ifinal);

        BWtest=imbinarize(I2test,BWthresh1);
        I2mid=imbinarize(I2mid,BWthresh);
        I2final=imbinarize(I2final,BWthresh);

        %Denoise with Weiner2 filter
        for (i=1:1:WF1)
            BWtest = wiener2( BWtest,[5 5]);
        end

        for (i=1:1:WF)
            I2mid = wiener2( I2mid,[5 5]);
            I2final = wiener2( I2final,[5 5]);
        end
       montage({I1test,Imid,Ifinal,BWtest,I2mid,I2final}, 'Size', [2 3]);
       str_caption = sprintf("F%d BW1%.2d WF1%.2d BW%.2d
WF%.2d",framenum,BWthresh1,WF1,BWthresh,WF);
       title(str_caption);
       %Export Frame
        expTail = strcat(filename,'_ImgSettings.jpg');
        expPath = strcat(dataExpDir,expTail);
        saveas(gcf,expPath);
    end
    disp('Step 4 complete: BW threshold set!');

%% Step 5 - Select Baseline

    disp('Step 5 Set Baseline...')
    %Choose a frame for baseline selection
    disp('     Choose Frame for Baseline Set')
    framebase=chooseFrame(framenum,frames,framesDir,filename,-1);


I1test=imread(strcat(framesDir,filename,'\',filename,'_',num2str(framebase),'
.jpg'));




    disp('     Choose Baseline Set')
    [~,row]=ginputc(1,'Color','r','LineStyle',':');
    baseline=round(row);
    disp('Step 5 Complete: Baseline set!')

%% Step 6 Choose Column and 1st frame to analyze height for
close all

    disp('Step 6 Set Column')
    %Choose a frame and click on image to select which column of pixels to
```

```
    %use. Or assign pixel column number in next section instead.

    disp('     Choose Column for Height MSMT')
    framenum=chooseFrame(framenum,frames,framesDir,filename,-1);



I1test=imread(strcat(framesDir,filename,'\',filename,'_',num2str(framenum),'.
jpg'));
    I2test=rgb2gray(I1test);
    colFrame=imbinarize(I2test,BWthresh1);
     %Denoise with Weiner2 filter
     for (i=1:1:WF)
            colFrame = wiener2( colFrame,[10 20]);

     end
    imshow(colFrame)
    [col,~]=ginputc(1,'Color','r','LineStyle',':');
    col=round(col);

    %Create labeled image
    str_caption = sprintf("F%d BW1%.2d WF1%.2d BW%.2d WF%.2d Col%d
Base%d",framenum,BWthresh1,WF1,BWthresh,WF,col,baseline);
    colFrame(:, col) = 255; % White = 255, can pick any intensity.
    colFrame(:, col-1) = 255;
    colFrame(:, col+1) = 255;
    colFrame(baseline, :) = 255;
    colFrame(baseline-1, :) = 255;
    colFrame(baseline+1, :) = 255;
    imshow(colFrame)
    title(str_caption);


disp('Step 6 complete: column set!')

%% Step 7 - Calibrate Length (optional)

    disp('Step 7 Calibrate Length...')

     %Ask user if length needs to be calibrated
     inp = input("     Calibrate Length? Y/N: ",'s');
     if(inp=='Y')
          lengthCalib=1;
          disp(     'Draw rectangular mask around reference length [left edge
first!]');

          %show the original image at current framenum

imshow(imread(strcat(framesDir,filename,'\',filename,'_',num2str(framenum),'.
jpg')));


          %Draw polygon mask
          %mx and my are the vertex positions of your mask
          %left click for points, right click to close loop
          %right click a vertex to create mask
```

```
            [mask, mx, my]=roipoly;

            refPix = ((mx(3)-mx(2))+(mx(4)-mx(1)))/2;

            %Display mask overlayed onto image
            imshow(imoverlay(mat2gray(colFrame),mask,[0.1 0.8 0.1]));

            refLength = input("     Enter Reference Length(um): ",'s');
            pSpeed = str2num(input("     Enter Printing speed(mm/min): ",'s'));
            mmPerPix = str2num(refLength)/1000/refPix;
            disp('Step 7 Complete: Length Calibrated!')
        else
            lengthCalib=0;
            disp('Step 7 skipped! - No Length param. set')
        end

           %Export Frame
           expTail = strcat(filename,'_maskEDFrame.jpg');
           expPath = strcat(dataExpDir,expTail);
           title(filename);
           saveas(gcf,expPath);

%% Step 8: Find meniscus height position for each frame
disp('Step 8: Obtain Meniscus Height at each frame...')

fstart=1; %start before needle appears, make sure col is at position before
needle passes
fspace=1;
fend=frames;
ypos=NaN([1,fend-fstart+1]);

%Early frames diff protocol
for k=fstart:fspace:20
    I1=imread(strcat(framesDir,filename,'\',filename,'_',num2str(k),'.jpg'));
    I2=rgb2gray(I1);
    BW=imbinarize(I2,BWthresh1);

    %Denoise with Weiner2 filter
    for (i=1:1:WF1)
       BW = wiener2(BW,[20 30]);
    end
    BWcol=BW(:,col);

    %height
    hpos = find(~BWcol,1,'first');
    if(isempty(hpos))
        ypos(k)=0;
    else
        ypos(k)=find(~BWcol,1,'first');
    end
    fprintf('On frame: %d out of %d processed.\n',k-fstart+1,frames-fstart);
end

if(k < 20)
        BW=imbinarize(I2,BWthresh1);
    else
```

```
end


for k=21:fspace:fend
    I1=imread(strcat(framesDir,filename,'\',filename,'_',num2str(k),'.jpg'));
    I2=rgb2gray(I1);
        BW=imbinarize(I2,BWthresh);

    %Denoise with Weiner2 filter
    for (i=1:1:WF)
       BW = wiener2(BW,[20 30]);
    end


    BWcol=BW(:,col);
    ypos(k)=find(~BWcol,1,'first');
    fprintf('On frame: %d out of %d processed.\n',k-fstart+1,frames-fstart);
end
disp('Step 8 complete: Meniscus height obtained!')

%% Step 9: Calculate meniscus profile
disp('Step 9: Calculate Meniscus Profiles...')

[~,minindex]=min(ypos);%get frame that needle first enters view

menheight=baseline-ypos; %substract baseline to get height of meniscus in
pixels

%mmPerPix=0.23/520;%real world length per pixel for conversion, from needle
width or another reference
if (lengthCalib==1)
    menheightreal=menheight.*mmPerPix;
end

%Average frame rate of video file for calculation. If frame rate is
inconsistent or
%there are dropped frames then you must find the time of each individual
%frame instead
framerate=obj.FrameRate;

%frame number array
xframe=1:fspace:fend;
%time array
xtime=(xframe-fstart)./framerate;%seconds

tOffset = minindex/framerate;%time needle first enters view
xtime = xtime-tOffset; %corrected time so 0 is when needle enters

%Use printing speed to scale distance
xDist = xtime*pSpeed*60; %mm

%Plotting
close all;

figure (1);

%Plotting realheight vs distance
```

```
    subplot(2,3,1);
    plot(xDist,menheightreal,'k','Linewidth',1.5)
    title('Meniscus Length')
    xlabel('Distance from Nozzle(mm)')
    ylabel('Height(mm)')
    set(gcf,'color','w');
    box on

%Plotting arb. axes vs frame
    subplot(2,3,4);
    plot(xframe,menheight,'k','Linewidth',1.5)

    %add thick vertical lines every 10/100/1000 frames
    if (frames > 1000)
        for i=0:100:frames
            xline(i,'k');
        end

    elseif (frames > 10000)
        for i=0:1000:frames
            xline(i,'k');
        end
    else
        for i=0:10:frames
            xline(i,'k');
        end
    end

    title('Raw Calculations')
    xlabel('Frame')
    ylabel('Height (pixels)')
    set(gcf,'color','w');
    hold off
    box on

    %Plot realheight vs time
    subplot(2,3,[2 3 5 6]);
    hold on
    plot(xtime,menheightreal,'k','Linewidth',1.5)
    xlabel('Time elapsed (s)')
    ylabel('Height(mm)')
    set(gcf,'color','w');
    box on
    hold off
    set(gcf, 'Position',  [100, 100, 1200, 500])

    %Export Figure 1
        expTail = strcat(filename,'_allPlots.jpg');
        expPath = strcat(dataExpDir,expTail);
        saveas(gcf,expPath);


    figure(2)
    hold on
    plot(xtime,menheightreal,'k','Linewidth',1.5)
    title(filename)
    xlabel('Time elapsed (s)')
```

```
    ylabel('Height (mm)')
    set(gcf,'color','w');
    box on
    set(gcf, 'Position',  [100, 100, 700, 500])
        %Export Figure 2
        expTail = strcat(filename,'_HvsT.jpg');
        expPath = strcat(dataExpDir,expTail);
        saveas(gcf,expPath);


    %Export Data to File

    %Concatenate results into matrix [frame xtime(s) xDist(mm)
menheight(pixel) menheightreal(mm)]
    m = [transpose(xframe) transpose(xtime) transpose(xDist)
transpose(menheight) transpose(menheightreal)];

    expTail = strcat(filename,'_Data.txt');
    expPath = strcat(dataExpDir,expTail);

    %Write data file headers
    fid = fopen(expPath, 'w');
    fprintf(fid, "Frame,TimeElapsed(s),Distance from nozzle
(mm),Height(pixel),Height(mm)\n");
    fclose(fid);

    %Write data
    dlmwrite(expPath,m,'-append','delimiter',',');
    disp('Step 7 Complete!')



    disp('Step 9 complete: Sequence Complete!')


%% Local functions
function pickedFrame = chooseFrame(frame, maxFrames, framesDir,
filename,mask)
    currentFrame = frame;
    pickedFrame = 0;
    done = 0;

    while (done ~= 1)
        %display masked, if mask given

        im
=imread(strcat(framesDir,filename,'\',filename,'_',num2str(currentFrame),'.jp
g'));

        if (mask~=-1)
            imshow(imoverlay(mat2gray(im),mask,[0.0 0.5 0.0]));
        else
            imshow(im);
        end
```

```
        cmd = input('Choose Frame: q for done, asd+fgh for -
(10/5/1)+(1/5/10), frames: ','s');
        switch cmd
            case 'q'
                done = 1;
                pickedFrame = currentFrame;
            case 'd'
                if (currentFrame > 1)
                    currentFrame = currentFrame-1;
                end
            case 's'
                if (currentFrame > 6)
                    currentFrame = currentFrame-5;
                end
            case 'a'
                if (currentFrame > 101)
                    currentFrame = currentFrame-100;
                end
            case 'f'
                if (currentFrame < maxFrames)
                    currentFrame = currentFrame+1;
                end
            case 'g'
                if (currentFrame < maxFrames-5)
                    currentFrame = currentFrame+5;
                end
            case 'h'
                if (currentFrame < maxFrames-100)
                    currentFrame = currentFrame+100;
                end
            otherwise
                disp('Unrecognized input\n');
        end

    end
end
```

§15. Solvent Vapor Annealing Study

The apparatus for annealing during *in situ* microscopy comprised a small glass petri dish sitting (base down) within a larger petri dish (also base down) on a hot plate set to the target temperature underneath a microscope camera.

In the absence of solvent vapor, structural rearrangement is not evident after several hours even at the highest modest temperature used (65°C). Into the small petri dish were placed a metal weight and the desired sample. An excess of THF solvent was injected into the larger petri dish and the system was partially capped with a glass petri dish cover (preheated to 150°C to avoid condensation) and held closed for 300 seconds. Then the dish is uncapped and the remaining THF solvent was removed via syringe pump from the reservoir and the sample could deswell. All experiments took place in a fume hood with strong forced convection.

Microscope camera images of solvent-vapor annealed samples discussed in the main text.

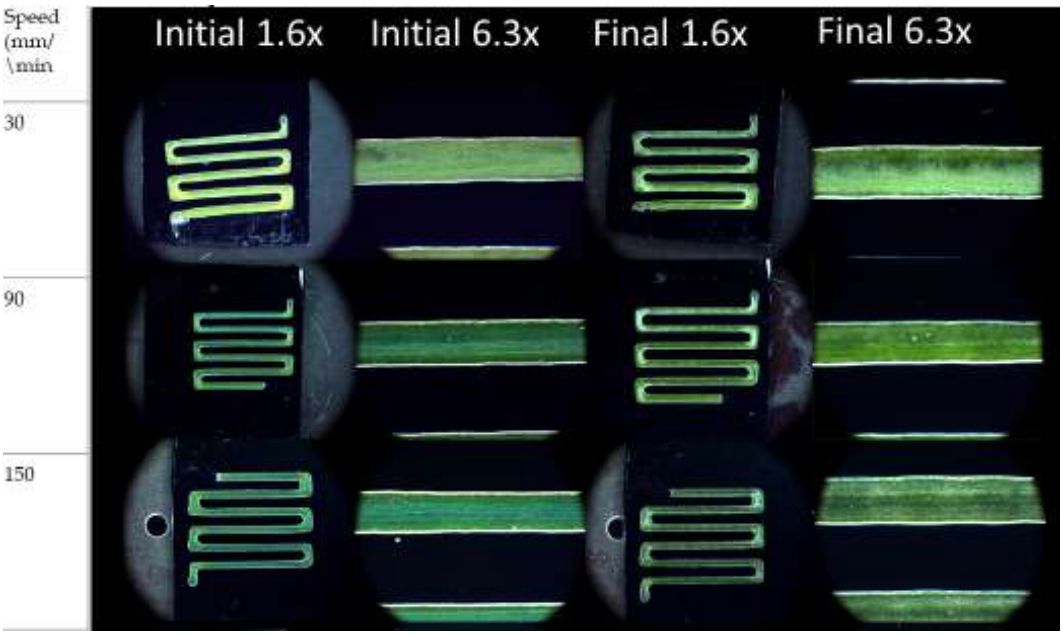


Figure S47 Printed at 25C, annealed at 25C.

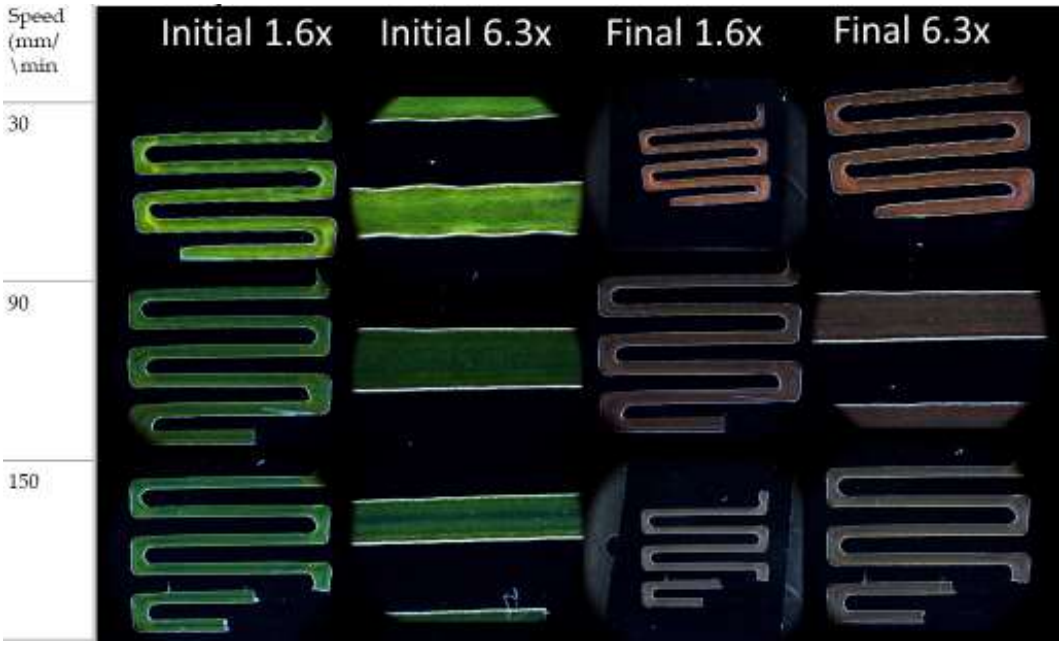


Figure S48 Printed at 25C, annealed at 50C.

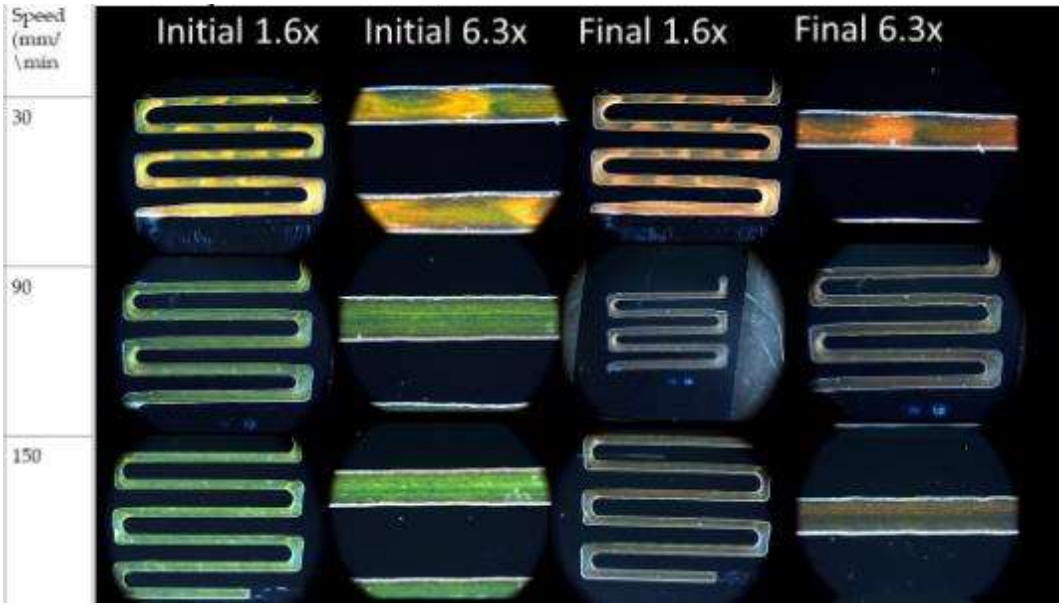


Figure S49. Printed at 50C, annealed at 50C.

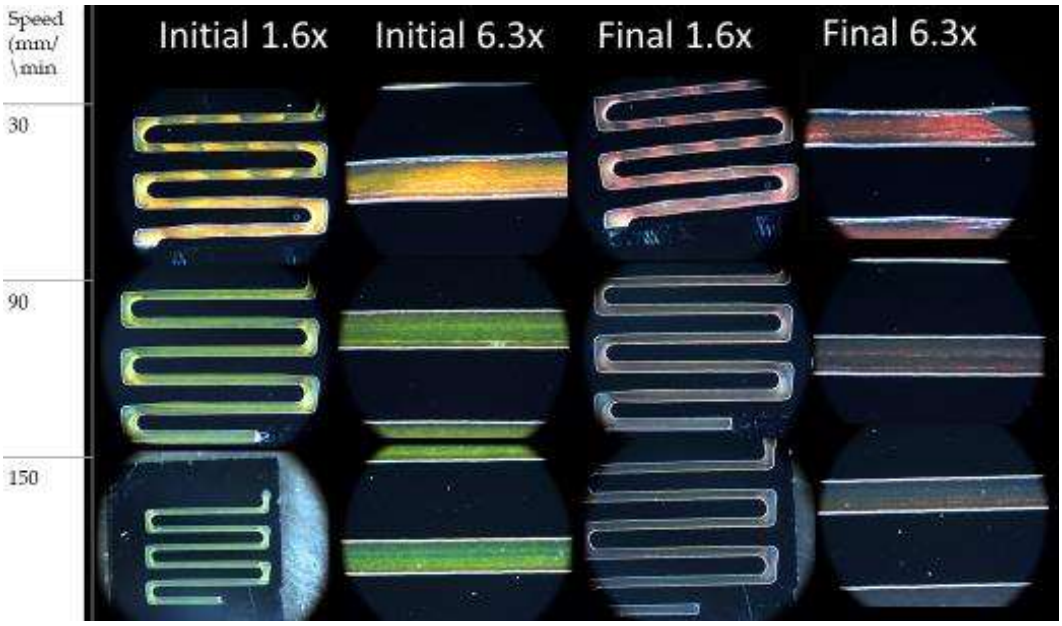


Figure S50. Printed at 50C, annealed at 65C.

Microscope Images of Dropcast Solvent Annealed Samples

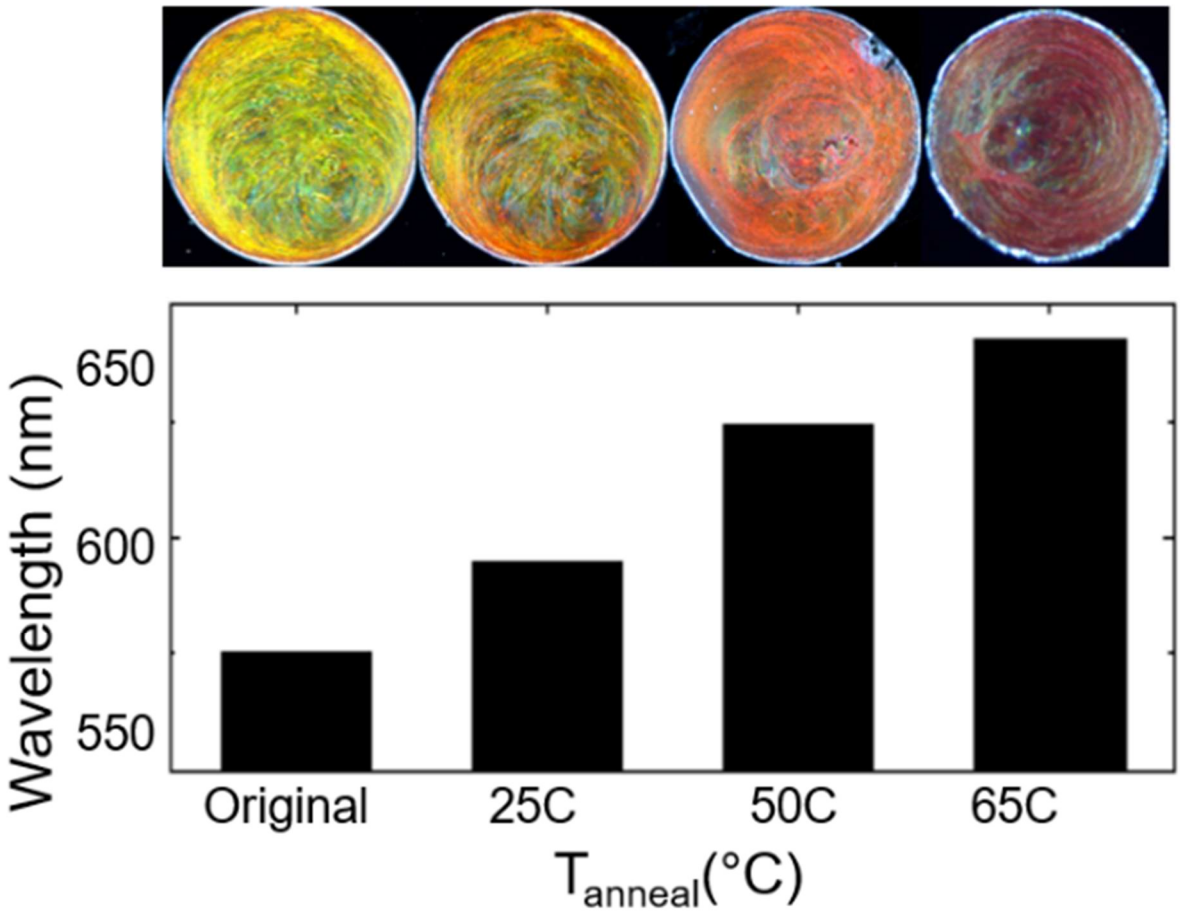


Figure S51. Images and peak reflected wavelength for dropcast samples prepared at 25 C and then solvent vapor annealed at the indicated temperatures.

§16. Thermal Annealing

The following result is from compression/annealing of the as-synthesized polymer between glass slides in a vacuum oven for 3 hours at 150 C. We see that after thermal annealing, the peak reflected wavelength is ~605 nm, (corresponding to optically estimated d-spacing of 212 nm, see SI page 40). This is well above that of the majority of our printed films and suggests that the printed films are indeed trapped in a metastable conformation.

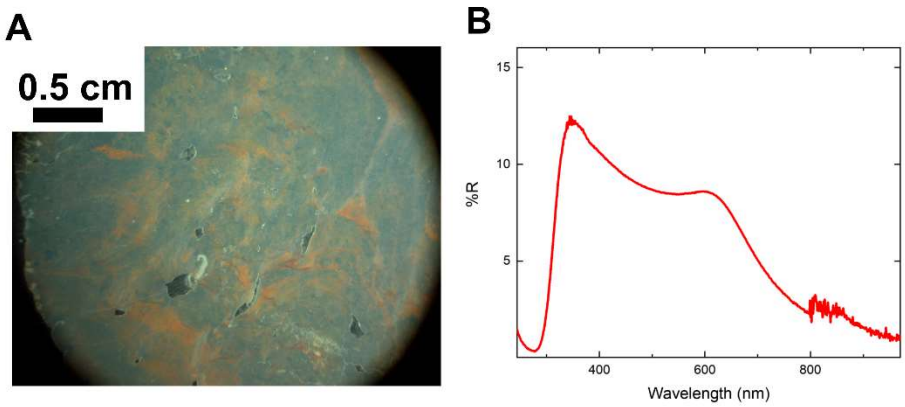


Figure S52. Thermal Annealing results. (A) Microscope camera image of annealed film under a ring light.