

Cross-Section and Top-View Image-Analysis Pipelines for Automated Characterization of 3D-Printed Microfluidic Channels and Valves

Huu Anh Minh Nguyen ¹, Mark Volosov ³, Dae Jung Martins Cruz ³, Abigail Goz ⁵, Maanit Khanna ¹, Rishabh Jitender ¹, Quynh Ai Vuong ¹, Nhu Doan Quynh Nguyen ¹, Bernardino Almanzar ³, Aidan Kinney ¹, Sergei Chapek ⁴ and Roman Voronov ^{1,2,3*}

¹Otto H. York Department of Chemical and Materials Engineering, Newark College of Engineering, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA.

²Department of Biomedical Engineering, Newark College of Engineering, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA.

³AltVIVO, Inc., Newark, NJ 07105, USA.

⁴Independent Researcher, Rostov-on-Don, Russia.

⁵Department of Computer Science, Ying Wu College of Computing, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA.

* Corresponding author: Prof. Roman Voronov. Email: rvoronov@njit.edu

Abstract

Printing a microfluidic device in one piece encloses its channels and valve membranes inside a solid body, where inspection from the outside cannot easily tell whether a channel formed or how far a membrane moves. In a companion study we built a design framework for printed membrane valves that rests on measurements taken from micrographs of purpose-built test devices. Here we report the imaging methods and the three open-source analysis programs that produce those measurements. Two of the programs analyze cross-section views, in which the microscope looks along each channel through its open end while light shines through the device from behind. In that view an open channel appears as a bright window, and a blocked channel or a deflected membrane darkens it. A general-purpose segmentation model (the Segment Anything Model) outlines candidate channel regions without task-specific training. A sequence of classical image tests then decides which candidate is the channel. The first cross-section program classifies each channel as open, occluded, or not formed and measures how far the channel roof sags. The second compares images of each valve before and after pressurization and fits a circular arc to the edge of the membrane to measure how far the membrane reaches into the channel. A set of quality checks turns an unreliable reach measurement into missing data rather than a false zero. The third program, still under development, analyzes top views taken through the transparent resin. The top-view program tracks both walls of every channel along its full length and classifies each channel as formed, narrowed, pinched closed, or not formed. Applied to 112 top views, it located narrowings and closures partway along channels, which a cross-section at the channel end cannot show, and it agreed with an expert on 17 of 24 reviewed channels. Each program also writes a diagnostic figure for every automated decision.

Keywords: 3D-printed microfluidics; image analysis; Segment Anything Model; zero-shot segmentation; membrane valves; printability; transmitted-light microscopy; channel tracking; quality control; open-source software

1. Introduction

Light-based additive manufacturing, namely digital light processing (DLP) and stereolithography (SLA), builds enclosed three-dimensional microfluidic networks in one piece and removes the alignment and bonding steps of layered soft lithography¹⁻³. That same one-piece construction creates a measurement problem. The channels, membranes, and valve seats that decide whether a printed device works sit inside a solid photopolymer body once printing ends. Profilometry can inspect a soft-lithography mold before assembly, whereas any inspection of a printed device has to look through the part itself. The standard options suit iterative design work poorly. Destructive sectioning sacrifices the device and can deform soft features during the cut. Micro-computed tomography resolves internal geometry without damage^{4,5}, but it is slow and costly for screening hundreds of design variants. Plain top-down optical inspection, finally, superimposes every layer along the optical path.

In a companion study⁶ we introduced a geometry-first framework for developing membrane valves in commercial resins that lack published mechanical data, building on earlier demonstrations of 3D-printed membrane valves and pumps^{3,7-9}. In place of mechanical properties, the framework uses two families of image-derived measurements. The first family comes from passive *printability devices*, enclosed rectangular channels spanned by thin printed roofs. The framework classifies each printed channel of such a device as open, occluded, or not formed, and it measures the sag of the channel roof. The second family comes from *functionality devices*, which add a pneumatic control chamber above the same channel so that the roof becomes a valve membrane. For each functionality device we measure how deeply the pressurized membrane reaches into the channel and how much of the channel it obstructs. A single constant extracted from these measurements, the membrane deflection coefficient κ , then parameterizes a geometric closure model and an interactive design tool. The framework therefore depends on the quality of these measurements. For NanoClear resin alone, they come from 990 printability images and 123 open/closed image pairs of functionality devices (see Section 3.5). Manual annotation at that scale would be slow. Manual annotation would also be irreproducible. The calls that matter most, such as whether a dim channel is occluded or where a membrane front ends, are subjective exactly at the feasibility limits the framework exists to find.

This paper describes the tools we built to make those measurements automatically, in three parts. The first part is an imaging method that turns the internal state of a device into optical contrast using a zoom stereomicroscope with a transmitted-light base. The test devices are designed as *cross-sectional mimics*: the front and back ends of each channel are left open, so each device presents its own cross-section to the microscope without cutting. Lit from behind, an open channel appears as a bright window, a partly filled channel darkens, and a deflected membrane shadows the window from above.

The second part consists of two analysis programs for these cross-section views: a sag/printability program and a valve-closure program. Both hand segmentation to the Segment Anything Model (SAM)¹⁰, a promptable model that outlines objects it was never trained on. Both then apply all domain knowledge after segmentation, as a sequence of classical image tests. A conventional supervised network, such as a U-Net¹¹, would have needed a labeled training set that did not exist and would need new labels for every device geometry, resin, or microscope. SAM, by contrast, supplies good candidate outlines without training, and the classical tests encode the physics of backlit imaging in rules that an expert can inspect one at a time.

The third part, new in this version, is a program for top views of the printability devices. A cross-section view samples one plane near the open end of a channel, so it cannot show a channel that is open at both ends but pinched shut halfway along. The top-view program fills that gap: it follows both walls of every channel from one end of the device

to the other and reports where and over what fraction of its length each channel narrows or closes. This program uses classical image processing only (see Section 3.6), and we are still refining it. We include it at its current stage of development, together with its first results (see Section 3.4).

Section 2 describes the imaging protocols and the algorithms of all three programs, including their parameter values and quality gates. Section 3 then presents representative outputs and lists the failure modes we hardened the cross-section programs against. It also reports the first top-view results and their agreement with expert calls, and it discusses how to organize the three programs. The source code of all three programs is available as described under Code Availability, and every panel in Figures 2 to 7 is output that the released programs generate themselves.

2. Materials and Methods

2.1. Devices and Fabrication

The devices imaged here are the two cross-sectional mimics of the companion framework⁶. They were generated parametrically in OpenSCAD and printed on an Asiga Ultra DLP printer (32 μm lateral pixel, 50 μm layer thickness) in FunToDo NanoClear resin under fixed manufacturer-recommended settings. Fabrication and post-processing are described in the companion manuscript⁶ and are summarized only briefly here. The *printability device* is a row of parallel enclosed rectangular channels, each spanned by a thin roof between solid side walls, with both ends of every channel left open. The *functionality device* adds a sealed pneumatic control chamber directly above the roof, which turns the roof into a membrane that pressure can deflect. Device dimensions are expressed in printer units. Channel width W is given in printer pixels (1 px = 32 μm), and channel height H and membrane (roof) thickness ML are given in printed layers (1 layer = 50 μm). The analysis programs inherit this convention directly (see Section 2.4).

2.2. Cross-Section Back-Illuminated Microscopy

We acquired all images on a Microquibic 3D MRCL700 zoom microscope (Microquibic AG, Zug, Switzerland) fitted with its transmitted (brightfield) illumination base. Cross-section views used the stock medium-zoom lens (lens 2), following the acquisition conventions of the companion study⁶ (see Figure 1A-C). The method relies on optics rather than computation. Because both ends of every channel are open, the channel is a through-cavity. Light entering from behind the standing device therefore crosses the channel almost unattenuated, while the surrounding cured resin absorbs and scatters it. The channel cross-section appears as a bright window bounded by darker material. When the channel narrows, fills with resin, or is shadowed by a deflected membrane, the window darkens accordingly. This first-order contrast tolerates modest focus and illumination errors, and it is what makes automated analysis feasible.

Each device stands on its end, and the microscope head views it almost horizontally. The microscope burns the tilt of the head into the footer of each frame, with 90° meaning horizontal. In a random sample of 20 cross-section frames that readout ranged from 89.4° to 96.7° (median 91.9°), and 18 of the 20 frames lay between 90.1° and 93.1°. This slight downward tilt keeps the near cross-section plane in view and separates it from the silhouette of the far open end. Because the devices are placed by hand, each frame also carries a small in-plane rotation, and the software corrects it automatically (see Section 2.7.2). Illumination also varies slightly from frame to frame, which the closure program compensates when it compares two images (see Section 2.8.1). The footer occupies the bottom 9% of each frame. The programs mask it from analysis and read the scale bar printed in it for calibration (see Section 2.10). Figures in this paper show the unmodified exported frames, and all quantitative analysis operates on them.

2.3. Top-View Microscopy

Top views of the printability devices were acquired on the same microscope with lens 1 and a head tilt of 0.00° , so the microscope looks straight down through the device onto the transmitted-light base (see Figure 1D). Each 1920×1200 px frame shows the whole row of channels of one device, with every channel running from the top face of the image to the bottom face. In this geometry the channel walls do not appear as a bright window. Instead, each wall appears as a thin dark line where the resin-air boundary refracts the transmitted light. The open channel ends show up as dark crescents or wedges at the two device faces, which we call *ports*. A collapsed roof resting on the channel floor appears as a closed dark outline inside the channel. The data set reported here comprises 112 top views of printability devices, 41 at $H = 5$ layers and 71 at $H = 10$ layers. Each device carries one design width, and each width has one to four replicate devices. A further 14 top views, taken earlier and reviewed channel by channel by an expert, served for tuning and validation (see Section 3.4).

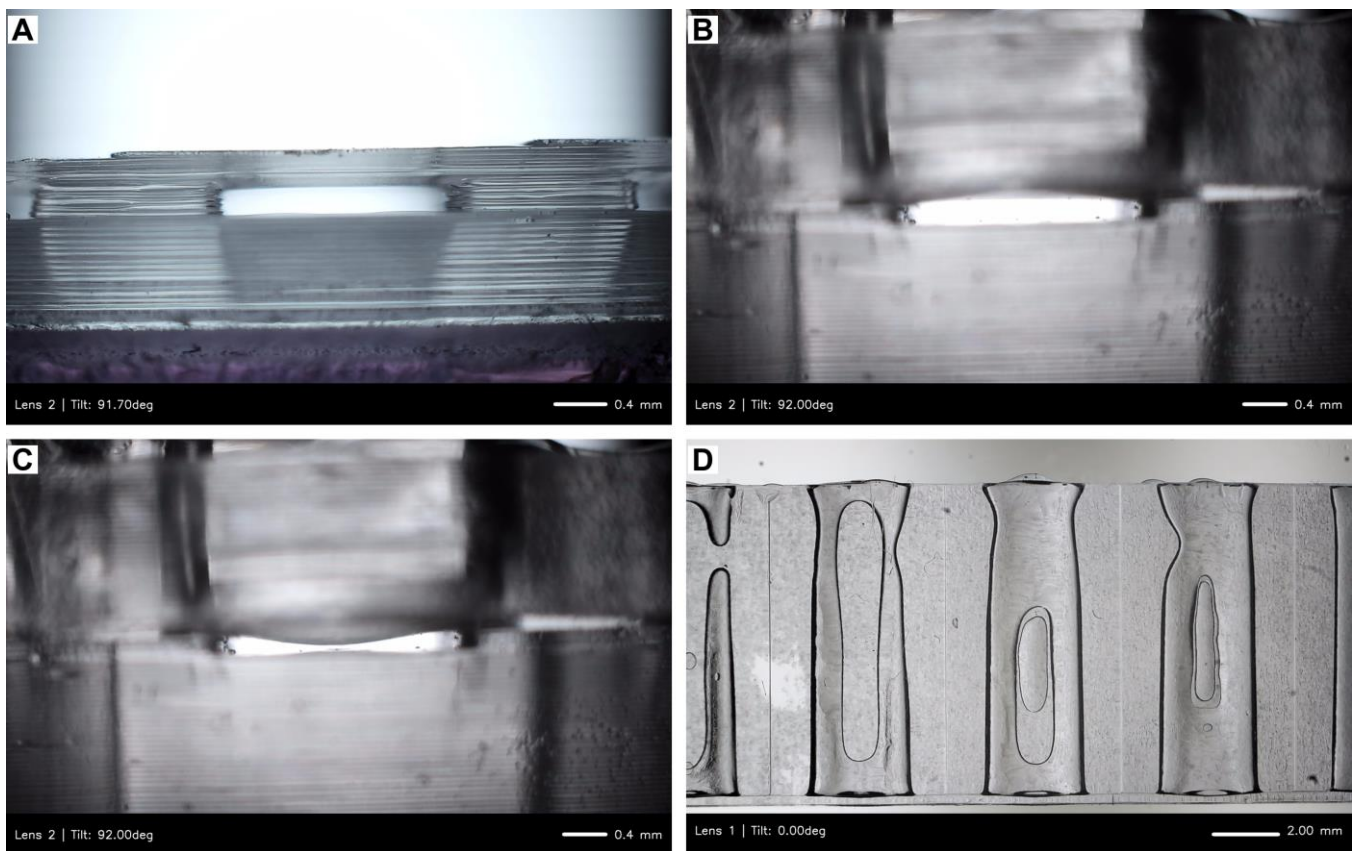


Figure 1 Raw frames as delivered to the three programs. (A) Cross-section view of a printability device ($H = 5$ layers, $W = 60$ px, $ML = 5$ layers): transmitted light enters through the open far end, so the open channel appears bright and the layered resin appears striated. (B, C) Cross-section views of a functionality device ($H = 5$, $W = 80$, $ML = 1$) with the control chamber unpressurized (B, open) and pressurized (C, closed). (D) Top view of a printability device ($H = 10$, $W = 100$): each channel runs from the top face to the bottom face, walls appear as dark refraction lines, and ports appear as dark crescents at both faces. The black footer carries the lens, the head-tilt readout, and the scale bar (0.4 mm in A-C, 2.00 mm in D).

2.4. Filename-Encoded Metadata and Batch Organization

All three programs treat the file system as an experiment database. Every acquisition parameter the analysis needs is encoded in the image filename, and each program derives its processing plan from a directory listing. Printability cross-section views follow the pattern $H<n>_W<n>_{ML}<n>_R<n>_{<MMDDYY>}$. Functionality cross-section views follow

where OP and CL denote the unpressurized (open) and pressurized (closed) valve state, MMDDYY is the experiment date (month, day, year), and redoN is an optional re-imaging tag. Top views carry _H<n>_W<n>_<replicate> anywhere in the name. The cross-section programs stop with an error on a filename they cannot parse rather than skipping it silently.

Three organizational behaviors follow from this convention. First, when several files share one experimental key, the highest-numbered redo wins, and the newest date breaks ties. A student can therefore drop a re-imaged frame next to a rejected original without deleting anything, and the program still selects the intended file. Second, the closure program joins the open and closed files on the full geometric key. It keeps a pair that lacks one state, scores it as zero closure because a valve imaged in only one state did not visibly actuate, and reports the missing image for follow-up. Third, before any slow processing starts, the printability program reconstructs the intended parameter grid from the data and reports under-replicated cells, missing combinations, and duplicates. Gaps in acquisition are therefore caught while the devices still exist to be re-imaged.

2.5. Zero-Shot Channel Segmentation with SAM (Cross-Section Views)

Both cross-section programs segment the channel with SAM¹⁰ (ViT-B backbone, checkpoint sam_vit_b_01ec64.pth; segment-anything 1.0, PyTorch 2.5.1 on the central processing unit (CPU), Python 3.10.11). MATLAB (R2026a) calls SAM through an out-of-process Python interpreter, so that a crash in the deep-learning runtime surfaces as an error MATLAB can catch rather than ending the session. The acquisition protocol centers the channel of interest in the frame, which reduces prompting to one convention. After the footer is cropped, one positive point prompt goes to the center of the remaining image, and the single mask SAM returns becomes the candidate channel. For context, the program then prompts a 6×6 grid of points spanning the central 80% of the frame, skipping points inside the candidate and keeping the best-scoring of each point's three proposals. It discards scene masks that overlap the candidate by an intersection-over-union above 0.5, that overlap each other above 0.7, or that SAM scores at 0.7 or below. The result is one candidate channel mask and a small library of masks covering the backlight and the printed material. That combination of masks is the input the validation sequence of Section 2.6 needs to decide, from the physics of the image, whether the candidate really is a channel. Raw SAM output is cached to disk per image (see Section 2.12), so inference runs at most once per micrograph however often the measurement code is revised.

2.6. Classical Validation of Candidate Masks

SAM proposes candidates, and a sequence of ten classical tests decides among them. Each test writes its own diagnostic figure (see Section 2.12 and Figure 2). (i) **Exclusive decomposition**: every mask is hole-filled, and pixels claimed by two masks go to the smaller one. (ii) **Speck cleanup**: Otsu thresholding¹² of the log-areas of each mask's connected components removes small disconnected fragments. (iii) **Texture scoring**: the program computes, over each mask's central interior, the density of Canny edges¹³ and of high local standard deviation (5×5 window, fixed threshold 0.03 on the normalized gray scale). The threshold is fixed rather than adaptive because adaptive thresholding amplified sensor noise inside smooth channels. Printed material is striated by layer lines and scores high, whereas open channels and backlight score low. (iv) **Backlight identification**: the backlight is the smooth, bright mask lying mostly above the channel, with a combined texture density of 0.5 or less. (v) **Anchor identification**: the strongly textured mask below the channel is the printed-material anchor, and tracing upward from the channel to the backlight gives the vertical extent of printed material. (vi) **Impossibility eliminations**: candidates touching the image border are removed, because an enclosed channel cannot touch it, and so are candidates centered outside the printed material. (vii) **Enclosure**

testing: grayscale morphological reconstruction scores how much each candidate behaves like an enclosed basin, either dark in bright surroundings or bright in dark surroundings. A candidate must beat the anchor's own score, which serves as the baseline for a region that is certainly not a channel. (viii) **Target-anchored selection:** a surviving candidate that contains the center point wins, with ties broken by enclosure score. Otherwise the nearest candidate wins. Near-ties, within 10% of the minimum distance, again go to the higher enclosure score. (ix) **Size gate:** the program rejects a selection larger than four times the channel area expected from the filename geometry and the image scale. The gate has no lower bound. An earlier lower bound rejected genuine, under-segmented dark channels instead of the specks it targeted. (x) **Contrast validation:** the median intensity inside the mask must differ from the median in a surrounding 25 px ring by at least 0.06 on the normalized gray scale. The sign of that difference assigns the polarity of the channel: bright (an open, transilluminated channel) or dark (a shadowed or occluded cavity). Bright masks must also match the nominal channel aspect ratio within a factor of 4, which rejects slivers of glare. Dark masks are exempt, because a channel that is starting to fill is physically flattened. A candidate that fails contrast validation is reported as a channel that did not form, never as a silent mis-segmentation.



Figure 2 Validation sequence: end-to-end diagnostic summary that the sag/printability program writes for one printability cross-section view ($H = 5$, $W = 60$, $ML = 5$). Panels 0-7, as titled in the figure: raw image; input with the center prompt (red cross); SAM mask returned for the center prompt; all scene masks after exclusive decomposition (colors identify individual masks); printed-material anchor; final validated channel mask with its area; channel overlay (green) with the assigned polarity; anchor (red) and channel (green) together. The text block lists the final status, polarity, SAM confidence, number of masks, texture contrast, and channel area. The program also writes one figure per validation step.

2.7. Sag/Printability Program (Cross-Section Views)

2.7.1. Channel-State Classification

Each printability cross-section view receives one of three states from the validation outcome. A validated bright mask is *open*, a validated dark mask is *occluded*, and a failed validation is *not formed*. Replicates then vote within each (width, roof thickness) cell, and a strict majority overwrites minority labels. Three physical rules then act on the map of states. (a) **Touchdown inference:** a not-formed cell wider than the last open cell in the same roof row is relabeled *membrane touchdown*, because at large widths the channel must have formed, so non-detection means the roof sagged onto the floor. (b) **Island removal:** a region of one label fully surrounded by another label takes the majority label of its neighborhood. (c) **Monotone banding:** within each roof row, the program enforces the physically required progression from not formed to occluded to open with increasing width, using the monotone relabeling that changes the fewest cells. The program also writes the maps before these rules, so the effect of each rule can be inspected.

2.7.2. Rotation Leveling and Sag Measurement

Because devices are placed by hand, each frame carries a small in-plane rotation. The program levels each image using the printed layer lines. Canny edges feed a Hough transform, lines within $\pm 30^\circ$ of horizontal are collected, and their median angle is removed by rotation whenever it exceeds 0.05° . The program then extracts the top edge of the leveled channel mask column by column, fills gaps, and smooths it with a moving median whose window is about $1/30$ of the channel span. A second-order polynomial fitted to that roof profile between the channel corners locates the sag apex at its vertex, with the profile maximum as a fallback and the fit R^2 reported. Sag depth is measured from two baselines in parallel. The *corner baseline* runs through the higher of the two channel corners, and the *bounding-box baseline* runs along the top of the smallest rectangle enclosing the mask. The program reports each sag depth in pixels, in millimeters, and as percentages of the measured height, the design height, and the channel span. The same pass reports channel geometry against the design (measured versus design width, height, and area), convexity, wall tilt, brightness statistics, and a debris estimate. A validity flag, requiring the sag vertex to lie between and below both corners, accompanies every measurement.

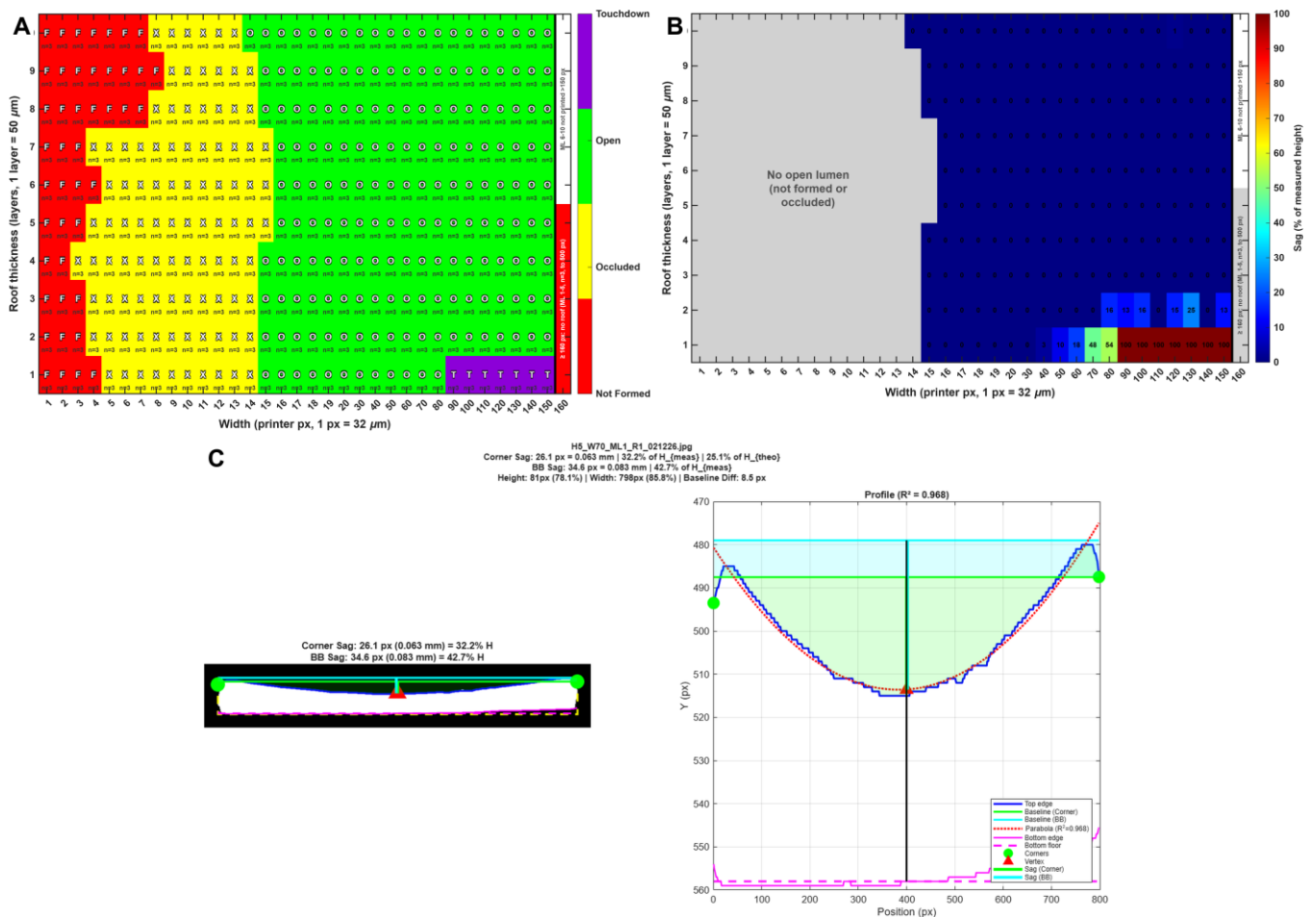


Figure 3 Sag/printability program outputs for NanoClear at $H = 5$ layers (990 cross-section views; $n = 3$ replicates per cell). (A) Channel-state map over width (printer pixels) and roof thickness (layers) after replicate voting and the map rules: F, not formed (red); X, occluded (yellow); O, open (green); T, inferred membrane touchdown (purple); n gives the replicates per cell. The column at 160 px marks widths printed without a roof (ML 1-5) or not printed (ML 6-10). (B) Roof sag at the channel midpoint as a percentage of measured channel height (color scale), measured from the corner baseline; gray marks cells with no open channel. (C) Diagnostic figure for one image ($H = 5$, $W = 70$, $ML = 1$): leveled channel mask with corners (green dots) and sag vertex

(red triangle), and the roof profile (blue) with the fitted parabola (red dotted), corner baseline (green), bounding-box baseline (cyan), and channel floor (magenta).

2.8. Valve-Closure Program (Cross-Section Views)

2.8.1. Open-State Reference and Registration

The closure program measures each valve from a pair of images, one unpressurized (OP) and one pressurized (CL). It segments only the open image with SAM. The validated open mask defines the reference channel footprint, the channel top in every column, and the open height and area that normalize every closure metric. The closed image is never segmented, because a nearly closed channel offers almost no coherent region to segment. Instead, the program compares the closed image to the open image photometrically inside the fixed reference footprint, which turns closure measurement into a differential problem. It first registers the closed frame to the open frame by phase correlation (translation only, computed above the footer). It accepts the estimated shift only if the shift raises the normalized cross-correlation with the open image above that of the unshifted image. This rule accepts genuinely large stage shifts and discards spurious estimates. The program then removes frame-to-frame illumination differences with an outlier-resistant linear mapping of the closed image onto the open image. It fits that mapping over the rows spanned by the channel, excluding a dilated band around the channel mask, and accepts a gain only between 0.5 and 2, with an offset-only correction as fallback.

2.8.2. Intensity-Loss Obstruction Mapping

Because the imaging is backlit, membrane material entering the channel blocks transmitted light, so the program reads closure as intensity loss. The loss map is the positive part of the open-minus-closed intensity inside the reference channel. The program thresholds it against a noise gate derived from the same image pair. The program estimates the standard deviation of the signed difference over the static reference region from the median absolute deviation. The gate is that estimate scaled to the 99th percentile of a Gaussian's positive tail and doubled, with a floor of 0.05 on the normalized gray scale. A pair counts as genuine zero closure unless at least 2% of channel pixels exceed the gate and at least 0.5% exceed twice the gate. The second criterion rejects diffuse edge glow, which covers area but lacks a strong core. A pair gated to zero that still shows a large response below the gate (at least 20% of channel pixels) is flagged as photometrically inconsistent and discarded as missing data. Surviving loss maps are normalized by the larger of the peak loss and the gate, so pure noise is never stretched to full scale. They are then thresholded by Otsu's method (capped at 0.15), consolidated morphologically, and filled column by column from the channel top to the deepest detected pixel to form the obstruction mask.

2.8.3. Membrane Reach by Model Fitting

Membrane *reach* is the deepest penetration of the membrane toward the channel floor, as a percentage of the open channel height. The program extracts it from the obstruction front, not from the raw mask. It traces the front only through pixels carrying at least 50% of the peak loss, because the membrane body blocks the most light while the dark gap beneath the apex does not. Without this restriction, reach would inflate toward 100% whenever the membrane approached the floor. The front depth is measured column by column over the central span of the channel, with 15% trimmed from each side. Columns next to the walls carry registration artifacts and are never used, even as a fallback. The profile is smoothed by a moving median and screened by an *edge cannot beat center* rule. The deepest point must lie in the central 40% of the span, and columns deeper than the center by more than a tolerance are voided. The cleaned profile, anchored at the channel walls, is then fitted with a one-parameter circular arc by bounded nonlinear least

squares (MATLAB lsqcurvefit, Optimization Toolbox). The arc is the same spherical-cap idealization that underlies the closure model of the companion framework⁶. The program accepts the arc at $R^2 \geq 0.5$, falls back to a wall-anchored parabola at $R^2 \geq 0.4$, and finally to the raw profile maximum, and it records which model it used. When a fit is accepted, the program recomputes the obstructed area by integrating the fitted profile, which makes the area immune to debris or fill under the membrane, and keeps the mask-based value as a diagnostic. The reported metrics per pair are area obstructed (% of open channel area) and membrane reach (% of open channel height), aggregated as mean $\pm$ standard error of the mean (SEM) over replicates.

2.8.4. Quality Gate: Discard, Do Not Zero

Every reach measurement then passes a multi-criteria quality gate. The guiding rule is that a bad measurement must become missing data, never a false zero. Zeros are reserved for validated no-deflection outcomes, which pass automatically. The gate discards a measurement if any of eight criteria fire: (i) the best fit has R^2 below 0.40; (ii) the fitted arc radius is below 80% of the smallest radius geometrically possible for the span; (iii) an edge peak exceeds the central maximum beyond tolerance; (iv) the mirror asymmetry of the depth profile exceeds 0.60; (v) the detected front spans less than 30% of the measurement span, since debris is narrow and membranes span the channel; (vi) area and reach disagree (area obstructed above 10% with reach under both 5 px and 3% of height); (vii) the obstruction does not attach to the channel top, since membranes descend from the roof and detached blobs are debris; or (viii) the fit is unphysical (non-positive arc depth, or an upward-opening parabola). The program compiles discarded and failed pairs into a re-imaging list that names the exact redo files to acquire (see Section 2.4). At the aggregation stage, one further rule, *trailing non-functional widths*, applies to each series. It drops widths with no trustworthy replicate above 1% reach only when they form a run at the wide end of a series. There the roof sags past the printability limit before any pressure is applied, so open and closed images are identical by construction. Genuine low-reach points at the narrow end are kept. The functional range of the companion study, up to 140 px with 150 px excluded as pre-sagged⁶, follows from this rule and the data rather than from a hard-coded limit.

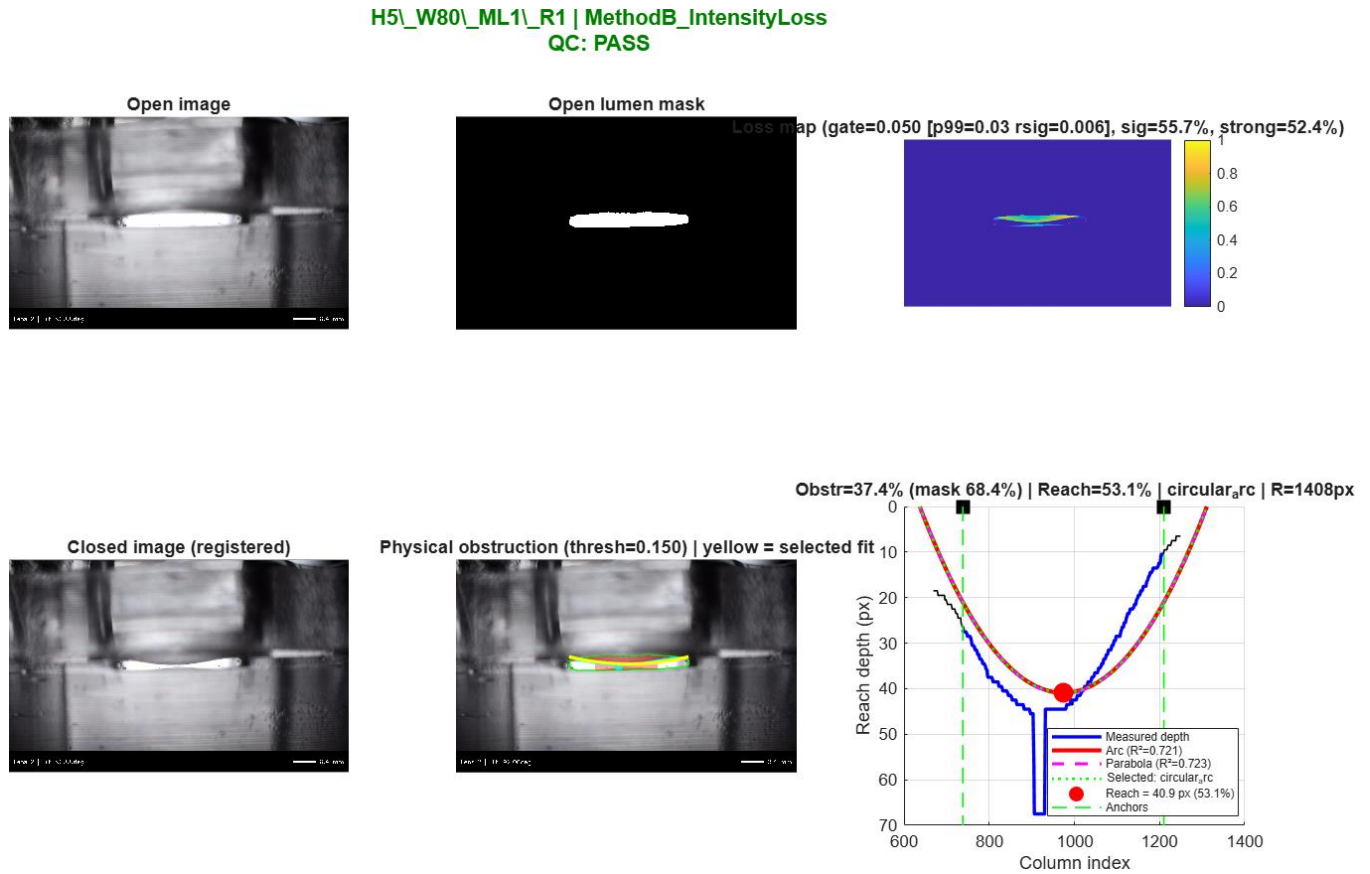


Figure 4 Valve-closure program: diagnostic figure written for every open/closed pair, shown for $H = 5$, $W = 80$, $ML = 1$, replicate 1. Top row: open image; validated SAM channel mask; normalized intensity-loss map, with the noise gate and the fractions of channel pixels above the gate (sig) and above twice the gate (strong) in the title. Bottom row: registered, photometrically normalized closed image; closed image with the reference channel outline (green), obstruction mask (red), and selected fit (yellow); reach profile with the measured depth (blue), circular-arc fit (red), parabola fit (magenta dashed), selected model (green dotted), wall anchors (green dashed), and the reach apex (red dot). A banner at the top gives the verdict of the quality gate (see Section 2.8.4).

2.9. Top-View Channel-Formation Program

The top-view program is a single Python script (Python 3.10; OpenCV 5.0, NumPy 2.2, SciPy 1.15, Matplotlib 3.10) that processes one image in 4 to 9 s. All of its parameters sit in one dictionary at the top of the script, each with a comment; lengths are fractions of the device height unless stated otherwise. The printer pixel pitch anchors the analysis. The program measures the scale bar in the footer, which is 2.00 mm long in every frame of this data set, and converts the design width W into an expected channel width in image pixels, $w_{\text{exp}} = W \times 32 \mu\text{m} / (\mu\text{m per image pixel})$. That expected width sizes the port filters, the tracking windows, and the channel lattice. The measured scale ranged from 9.8 to 14.3 μm per image pixel across sessions, so the program reads it per image. One image with an unreadable bar falls back to 13.5 μm per pixel, typical of these frames, and carries a flag.

2.9.1. Device, Maps, and Ports

The program first locates the device on a local-texture map (local standard deviation over local mean), which is insensitive to the illumination gradient across the frame. It fits lines to the top and bottom device edges and rotates the image to remove their mean tilt. It then divides the image by a heavily blurred copy of itself (Gaussian $\sigma = 12\%$ of the device height) to flatten the illumination. Three response maps follow, each scaled so that a value

of 1 means 30% darker than the local mean. Black top-hat filtering with a horizontal structuring element yields a *line map* of thin vertical dark lines (channel walls), filtering with a vertical element yields an *arc map* of horizontal dark edges (crescent rims and closed ends), and a *blob map* marks broad dark regions. Ports are short dark components touching either device face, searched within the outer 16% of the device height after lines taller than 35% of the height are removed. Column profiles of these components give port intervals. An interval wider than $1.6 w_{\text{exp}}$ is split at its intensity minima, because two neighboring openings have merged. Fragments closer than $0.25 w_{\text{exp}}$ merge, provided the merged opening stays below $1.3 w_{\text{exp}}$. For designs narrower than 12 image pixels the port is a small wedge, and absolute width limits replace the relative ones.

2.9.2. Channel Lattice

Ports on the two faces are paired into channels by proximity. The design pitch of the device is the solid wall between channels (2.0 mm, measured on seven devices) plus the channel width. The program accepts a pitch measured from the port pairs only when at least three pairs support it and it lies within 25% of the design pitch. Otherwise it uses the design pitch, phased on the strongest port pair. Fixing the pitch this way removed most channel-count errors on devices with faint or missing ports. Lattice slots without a port pair receive a center from a nearby single port or, failing that, from the lattice alone, and such channels are marked as predicted.

2.9.3. Wall Tracking

Within each channel's window the program tracks the left and the right wall from the top face to the bottom face by dynamic programming. Each track is the path that maximizes the summed response of the smoothed line and blob maps, less a penalty of 0.06 per pixel of lateral step, with steps limited to 2 pixels per row. At both faces a penalty of 0.30 per pixel draws the track toward the edges of the detected port. A soft penalty holds each track on its own side of the channel center line. Once one wall is tracked, the program suppresses the response near it so the second track cannot follow the same line. It runs the pair in both orders, left then right and right then left, and keeps the pair with the higher mean response, provided the two tracks stay at least 3 px apart over at least 30% of the rows.

2.9.4. Presence, Gaps, and Classification

A tracked wall counts as present only where it follows a continuous line. In a sliding window of 5% of the channel length, at least 75% of the rows must carry a response above 0.35, the width must vary smoothly, and the present run must connect to a port. The 75% threshold separates real outlines from speckle, which fills 40% to 50% of such a window. An interior gap in presence then receives one of two readings. The gap is a *closure* when the channel tapers below 40% of the reference width on both flanks of the gap, or when a dark cap spans at least 80% of the channel width at the gap end, as at a rounded blind end or where two tips converge. Otherwise it is a *tracking dropout*, and the program bridges it by interpolating the width across the gap. A closed run shorter than 6% of the length, where only one wall deflects, is treated as debris and ignored.

The *reference width* of a device is the median, over its well-tracked channels, of the 85th percentile of each channel's open width. That reference is the printed width, not the design width. The distinction matters, because at $H = 5$ the $W = 50$ channels print at about 40% of their design width, so thresholds relative to the design width would call every channel narrowed. Each channel then receives one of five classes. (i) **Formed**: open from port to port, with the narrowest point at 75% or more of the reference width. (ii) **Necked**: open from port to port but narrower than 75% of the reference width somewhere, which is a partial pinch. (iii) **Pinched**: closed somewhere, meaning narrower than 15% of the reference width. A channel that opens from one port only and ends inside the device is

pinched and flagged as *blind*. (iv) **Not formed**: nothing beyond the port wedges, or open over less than 20% of the length. (v) **Unresolved**: both walls lost over more than half the length without closure evidence, which calls for review by eye. Alongside the class, the program reports the narrowest width and its position, the fraction of the length that is narrowed or closed, and where the defect lies. It also reports *touchdown*, the fraction of the channel area enclosed by an inner closed outline, which marks the roof resting on the floor.

Two confidence levels accompany the classes. A channel has low confidence when a bridged dropout exceeds 30% of its length. A device has high confidence with three or more aligned port pairs, or with one pair and two well-tracked channels. A device has low confidence when no port pair exists, or when the reference width falls below 35% of the design width, which means the tracks followed walls rather than channel edges. Summaries per design exclude low-confidence devices. The program writes one table row per channel, one per device, and one per design (H, W), plus one diagnostic overlay per image (see Figure 6).

2.10. Calibration and Units

The programs use three unit systems deliberately. Design dimensions arrive from filenames in printer units (32 μm per lateral pixel, 50 μm per layer), the same convention used to generate the devices^{6, 14}. Image scale comes from the scale bar in each frame's footer. The cross-section programs locate the millimeter label by optical character recognition (OCR), extract the bar beside it, and read the label value. They trust that value only within a factor of two of the 0.4 mm bar used for every cross-section view in this data set and substitute 0.4 mm otherwise. The substitution was common: OCR returned no value for 620 of the 990 printability images and most often dropped the decimal point in the rest. The top-view program measures the bar length in pixels and takes its length as 2.00 mm for every frame. The H = 5 top views carry a 0.20 mm label on a bar of the same pixel length as the 2.00 mm bars of the H = 10 views, and only 2.00 mm reproduces the design widths of the W = 50 and W = 100 channels, so the 0.20 mm label is a calibration slip in the microscope software. The headline metrics of all three programs, however, do not depend on calibration. Sag, reach, and area obstructed are percentages of dimensions measured in the same image, and the top-view classes are fractions of each device's own printed width. Scale errors therefore cancel, and images from different zoom sessions remain comparable. Physical units enter the published results only where the companion framework requires them, in the conversion of the reach-versus-width fits into the membrane deflection coefficient κ , and there only through the printer-unit constants.

2.11. Aggregate Model Fitting

The closure program ends by fitting the linear deflection models that the companion framework consumes. For each (height, membrane thickness) series it fits replicate-level reach two ways. The *onset model*, $\text{reach} = k(W - W_0)$, yields the slope k and the closure-onset width W_0 . The program fits it over the rising part of each series, selected automatically as the longest contiguous run of at least three widths whose mean reach lies strictly between 3% and 85%. The lower bound excludes onset noise, and the upper bound excludes saturation at floor contact. The *through-origin model* in physical units, $s = \kappa C$, where s is membrane sag and C is membrane width, yields the dimensionless κ that the design tool uses⁶. The output figures distinguish fitted from excluded points, and all fit parameters, ranges, and R^2 values go to CSV files with the per-pair and per-condition tables.

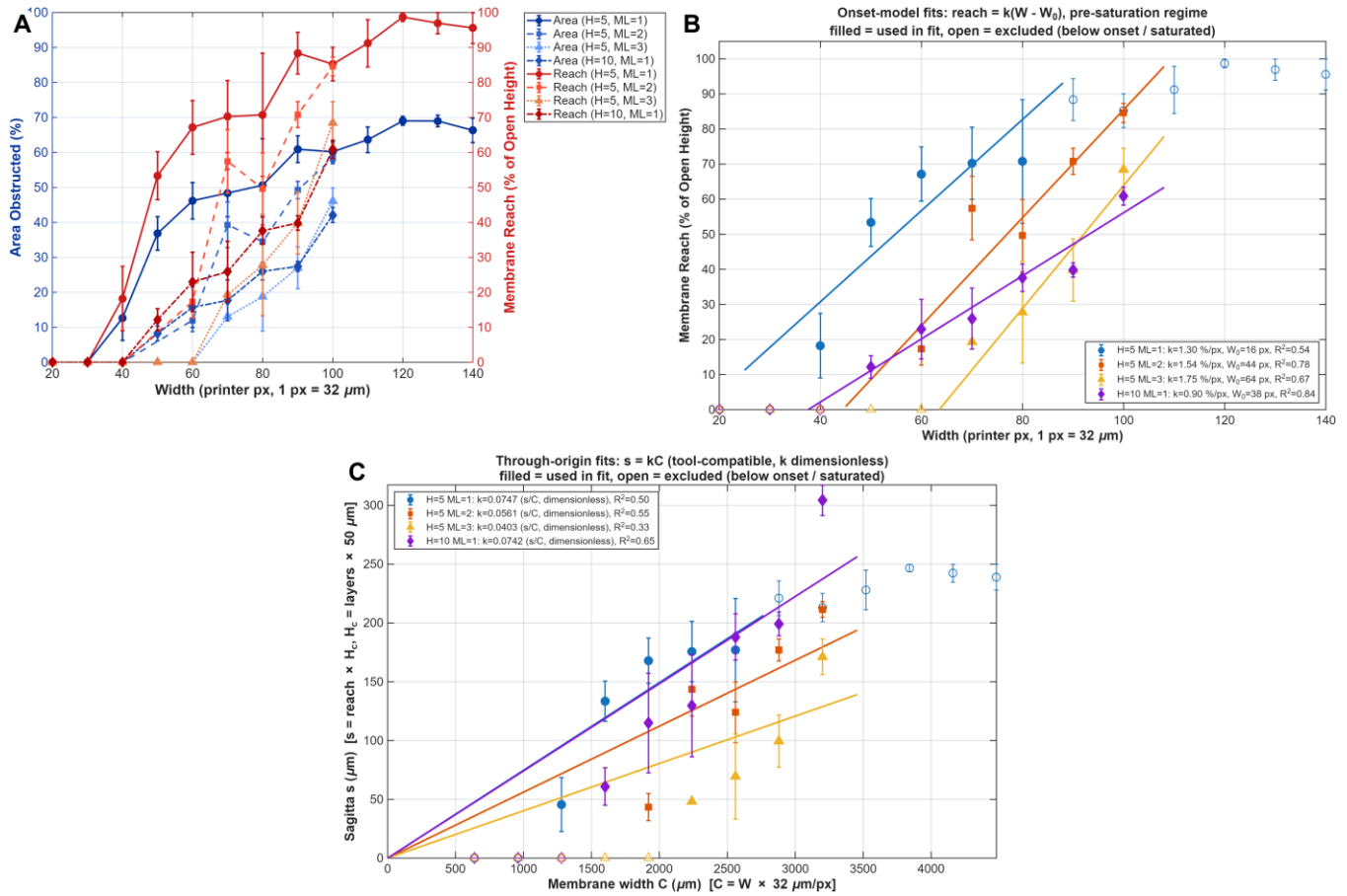


Figure 5 Valve-closure program: aggregate outputs for NanoClear. (A) Area obstructed (blue, left axis) and membrane reach (red, right axis) versus channel width for four series ($H = 5$ with $ML = 1, 2, 3$; $H = 10$ with $ML = 1$), mean $\pm$ SEM over replicates, after the trailing non-functional width rule. (B) Onset-model fits, $reach = k(W - W_0)$, over the automatically selected rising regime; filled markers were used in the fit and open markers were excluded (below onset or saturated). (C) Through-origin fits, $s = \kappa C$, in physical units, giving the dimensionless membrane deflection coefficient κ of each series.

2.12. Software Engineering: Caching, Diagnostic Figures, and Fault Tolerance

Three engineering patterns made the cross-section programs practical. (i) **Caching separated from measurement**: raw SAM output is cached per image as soon as inference finishes, in a directory outside the results tree, so that clean restarts keep it. OCR calibration is cached the same way. Complete per-image results carry a version stamp and are invalidated automatically when the code version, the table schema, or the file list changes. Almost all development happens after segmentation, so retuning a threshold and reprocessing the data set does not repeat inference. (ii) **Diagnostic figures as the human interface**: every validation step and every measurement writes a labeled figure, about a dozen per image plus the one-page summary of Figure 2. An expert therefore reviews the program's reasoning and not only its answers, and these figures, not summary metrics, exposed every failure mode in Section 3.3. (iii) **Fault tolerance for unattended runs**: an exception guard around each image keeps one bad frame from ending an overnight run. The deep-learning runtime runs in its own process, progress is saved every few minutes, and the program announces completion or failure by e-mail with a push-notification fallback. The programs write every table as a comma-separated values (CSV) file and a MATLAB data (MAT) file, and every figure as raster and vector graphics. The top-view program follows the same principle at smaller scale: it writes one overlay per image and one table

row per channel, and its CSV tables carry the underlying widths and fractions, so thresholds can be moved without re-tracking.

3. Results and Discussion

3.1. Imaging Performance

Backlit cross-section imaging carried the whole system, because every later algorithm uses contrast that the optics provide. Open channels appear as bright windows against striated material at every tested width, from 1 to 150 printer pixels (32 μm to 4.8 mm; see Figure 1A). Occlusion appears as graded darkening well before a channel closes geometrically. A pressurized membrane shadows the bright window with a sharply bounded front, whose intensity loss in functional devices lies far above the noise gate of each pair (see Figure 4). The remaining imperfections of the protocol are small in-plane rotations from hand placement, drift in illumination between frames, and occasional stage shifts between the two exposures of a pair. The software compensates each of them, by leveling, photometric normalization, and correlation-checked registration, respectively. Top views provide a different kind of contrast (see Figure 1D). The channel walls appear as thin dark refraction lines rather than as a bright area, which makes tracking lines, not segmenting regions, the natural analysis.

3.2. Why Zero-Shot Segmentation Plus Classical Validation

Raw SAM output alone does not segment these images correctly. Prompted at the channel, it can return slivers of glare, the channel merged with the backlight, or fragments of striated material, and it has no notion of which mask should be the channel. SAM does, however, almost always return the correct channel somewhere among its masks, across the full range of appearances from wide open and bright to nearly closed and dark. A validation sequence can exploit that property, whereas a network trained from scratch would have needed a labeled set spanning the same range. The division of labor is therefore clear. SAM contributes shape completion learned from web-scale data. The validation sequence contributes the domain physics as individually testable rules: backlight is bright, smooth, and above; printed material is striated and below; channels are enclosed, central, and contrast with their surroundings. Two lessons from building the sequence generalize. First, validation should rest on physics rather than on learned statistics. An earlier gate based on the ratio of texture densities between channel and anchor proved unable to separate the classes in expert review, and the simple signed contrast test replaced it. That test also assigns the bright/dark polarity that the printability classification uses directly. Second, rejections should be conservative. We removed the lower area bound once review showed that it rejected genuine dark channels. The surviving gates reject only physically impossible candidates, and scoring, not exclusion, resolves the remaining ambiguity.

3.3. Hardening Against Failure Modes

The closure program reached its final form only after a review of the per-pair diagnostic figures across the full data set exposed a series of failure modes. Each one is now handled by a specific guard. We list them because they are likely to recur in other backlit-imaging pipelines. (1) **Noise amplification**: normalizing a pure-noise loss map stretches it to full scale and turns a valve that did not close into an apparent near-total closure. The guard is the absolute noise gate with its dual criterion (see Section 2.8.2). (2) **Illumination mismatch** between the open and closed exposures masquerades as intensity loss across the whole channel. The guards are the photometric normalization and the discard of inconsistent pairs. (3) **Edge glow** along the blurred bright roof covers area without a strong core, and the strong-core criterion rejects it. (4) **Pinch-gap inflation**: the dark gap between a nearly touching membrane and the floor reads as loss and drags reach toward 100%. Tracing the front only through strong-loss pixels prevents it. (5) **Registration spikes** at the

channel walls produce single-column outliers in depth, which the trimmed span and the center-deepest rule remove. (6) *Fill under the membrane* corrupts the mask-based area, so the program recomputes area from the fitted membrane profile. (7) *Rejected large shifts*: a fixed limit on the registration shift silently discarded correct large translations. Residual-based checks could not catch the error, because the horizontal layer lines make them nearly blind to horizontal shifts. The guard accepts whichever transform, estimated or identity, correlates better with the open image. The terminal quality gate then catches debris, detached artifacts, and asymmetric fits (see Section 2.8.4).

The map rules of the printability program play the same role at the level of classification. One negative result is worth recording. We tried a statistical re-clustering of the per-image states across the data set (Otsu thresholding of the enclosure scores) and removed it after it erased a genuine occluded band in a full run. Per-image physics with replicate voting outperformed statistics over the data set.

3.4. Top-View Results: What a Single Cross-Section Misses

Figure 6 shows two top-view overlays. The first device ($H = 5$, $W = 50$) contains all three open-channel outcomes side by side (see Figure 6A). Channel 2 is formed from port to port. Channels 1 and 3 are open but narrow to 68% and 62% of the reference width. Channels 4 and 5 are closed over 31% and 25% of their length, respectively, and channel 5 closes at two separate points. A cross-section view taken at either open end of channels 4 or 5 would show an open channel, and so would a cross-section view of channels 1 or 3. The top view therefore adds information that the cross-section cannot provide. The second device ($H = 10$, $W = 100$) shows formed channels that contain closed inner outlines, where the roof rests on the floor over part of the channel while the channel stays open along both walls (see Figure 6B). A cross-section at the channel end would record such a channel as open.

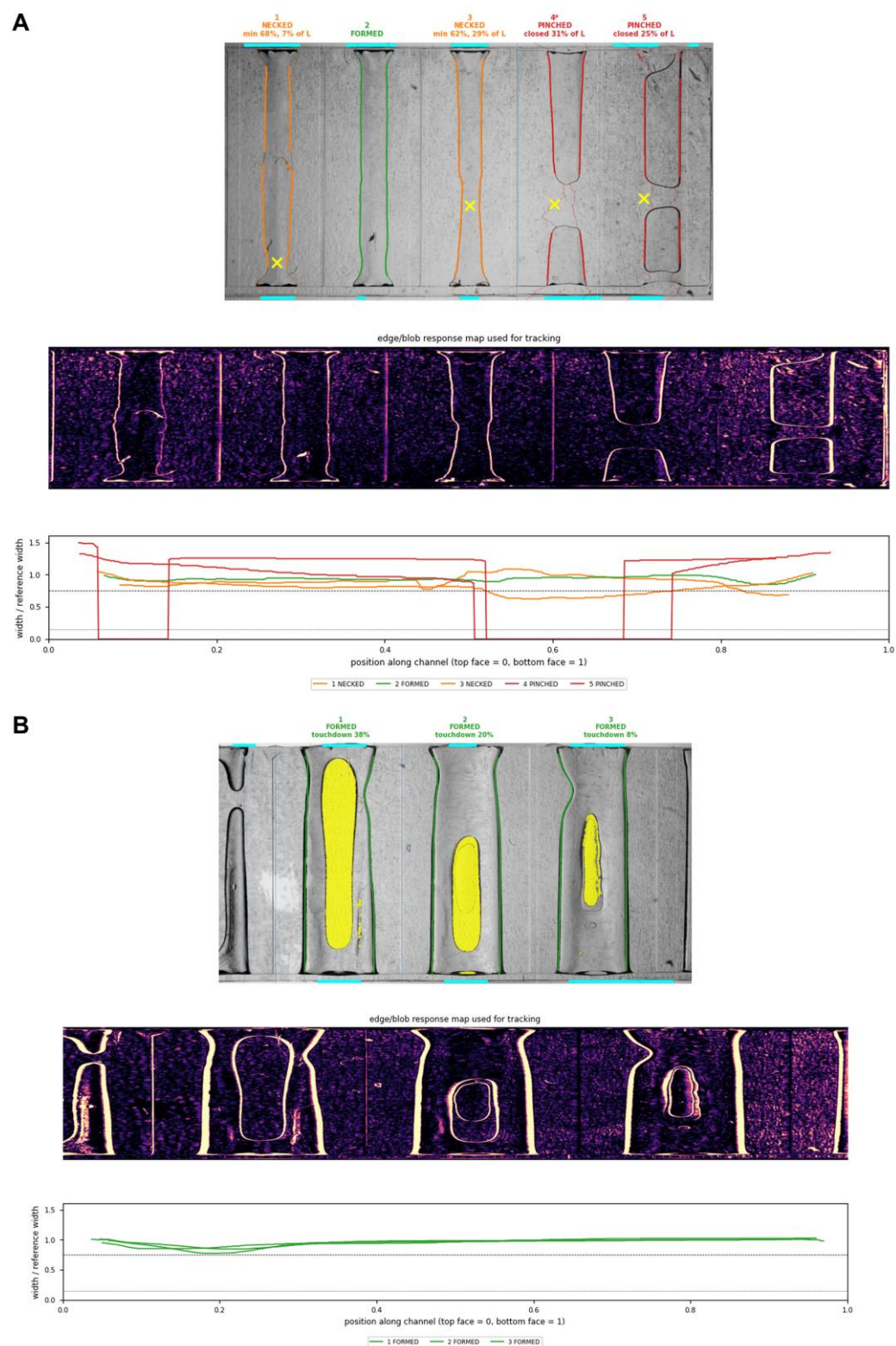


Figure 6 Top-view program: diagnostic overlays for two printability devices. (A) $H = 5$ layers, $W = 50$ px, from the expert-reviewed set. (B) $H = 10$ layers, $W = 100$ px. Top panel of each overlay: tilt-corrected device with the tracked walls drawn in the class color (green, formed; orange, necked; red, pinched; gray, not formed; cyan, unresolved), solid where present and faint where bridged; cyan bars, detected ports; dotted blue lines, walls between channels; dashed white lines, tracking windows; yellow $\times$, location of the narrowest point or closure; yellow shading, touchdown areas (inner closed outlines). Labels give the channel index (an asterisk marks a channel placed by the lattice), the class, and the narrowest width (% of the reference width), closed length (% of the channel length), or touchdown area. Middle panel: response map used for tracking. Bottom

584

585

586

587

588

589

590

591

592

panel: channel width divided by the reference width along the channel, with dashed lines at 0.75 (necked threshold) and 0.15 (closed threshold).

Across the full data set, the program classified 257 channels in 37 devices at $H = 5$ and 346 channels in 63 devices at $H = 10$, after excluding 0 and 0 low-confidence devices, respectively (see Figure 7). At $H = 5$, 70% of the channels did not form, 14% were pinched, 10% were necked, and only 4% formed from port to port. At $H = 10$, the corresponding shares were 23%, 31%, 19%, and 19%. The width dependence differs between the two heights. At $H = 5$, the narrow designs ($W \leq 20$ px) did not form in 75% to 100% of their channels, and the few exceptions were short blind channels open from one port. The $W = 30$ and $W = 50$ designs opened from end to end in two thirds of their channels, but most of those channels were necked, and the remaining third were pinched. The $W = 70$ design had the largest formed share (33%), a figure that Section 3.7 qualifies, while every channel of the $W = 110$ design failed to form. At $H = 10$, designs up to $W = 9$ px did not form, and designs from $W = 10$ to $W = 20$ px were mostly pinched, typically as channels open from one or both ports but closed over a third to a half of their length. From $W = 30$ to $W = 80$ px, the share of channels open from end to end rose from 58% to 92%, still with many necked channels. The $W = 90$ design formed in all 11 of its channels, and every channel of the $W = 100$ design was open from end to end. Above $W = 110$ px, formed shares fell again to between 18% and 71%, and pinched and unresolved channels returned. Touchdown concentrated in the $W = 90$ and $W = 100$ designs at $H = 10$, in 5 of 11 and 7 of 14 channels, respectively. These channels are open along both walls while the roof rests on the floor over part of their length, as in Figure 6B.

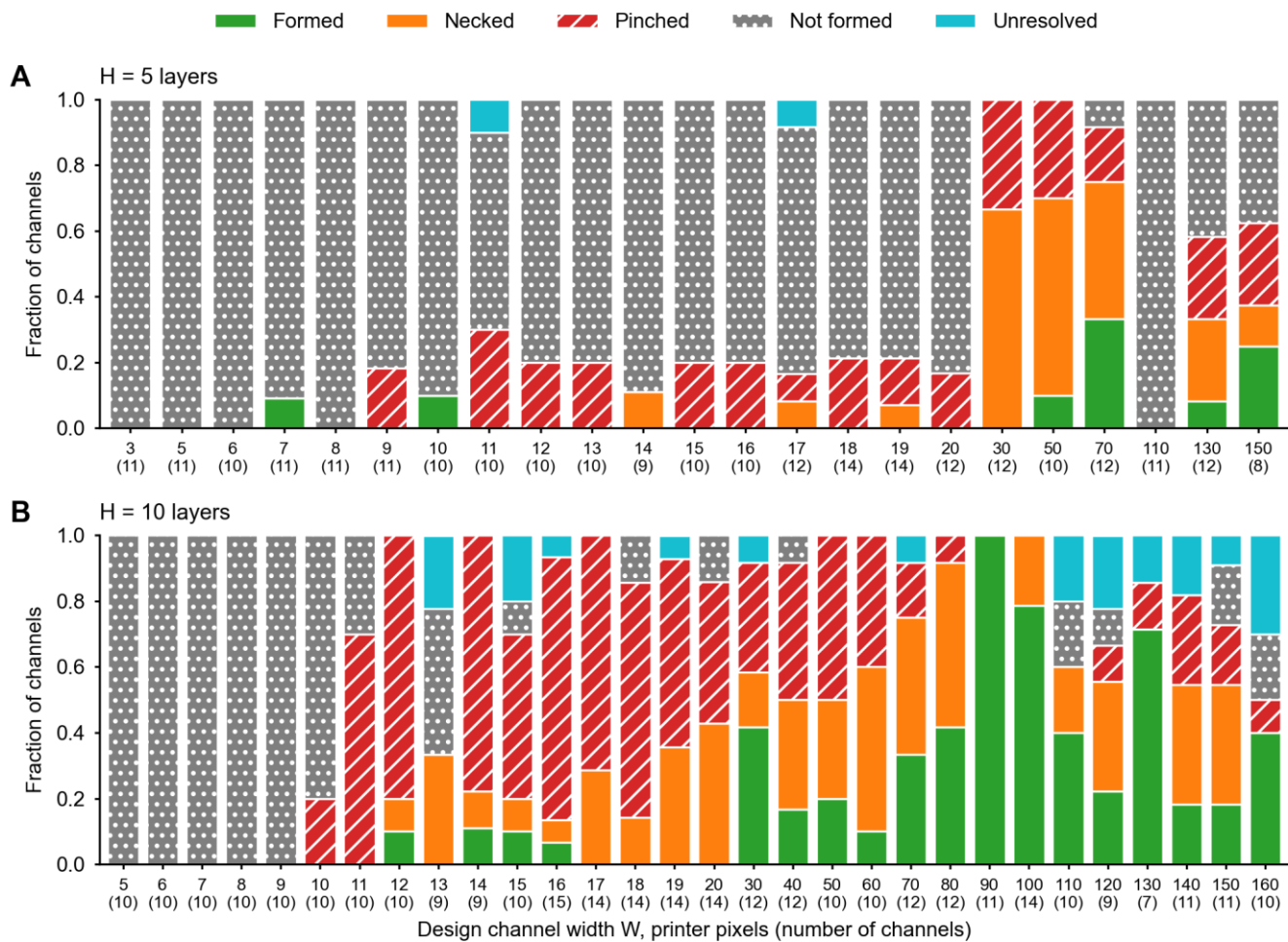


Figure 7 Top-view program: fraction of channels in each class versus design width for NanoClear printability devices at (A) $H = 5$ layers and (B) $H = 10$ layers. Each bar pools all channels of all devices of one design width; the number of channels is given in parentheses under each width. Devices with low confidence are excluded. Colors follow Figure 6; pinched and not-formed segments are also hatched. The width axis is categorical, not proportional.

To check these calls, we compared the program with an expert who had classified the channels of five of the 14 tuning images one by one, in the categories formed, partially pinched, and fully pinched. We counted a partial pinch as necked and a full pinch as pinched. On that basis the program agreed with the expert on 17 of 24 channels (71%). The seven disagreements run in both directions. In three of them the program called a channel pinched that the expert saw as open, once because one wall was faint, once because a port at the edge of the frame went undetected, and once because a tracking dropout where the channel flares into its port read as a closure. In two others the program called necked channels that the expert saw as fully pinched, once where two converging tips stopped 9 px apart and once where debris met a faint wall. One channel sat at the necking threshold, with a narrowest width of 68% against the 75% limit, and the program left one channel with a faint wall unresolved. The program also found one of the two touchdowns the expert marked, and it reproduced the expert's channel count on a wide device whose roof had collapsed along every channel. A second, independent expert set is needed before these figures can be read as an accuracy, a point Section 3.7 returns to.

3.5. Throughput and Human Effort

The printability run reported here processed 990 cross-section views: 10 roof thicknesses, 33 widths, and 3 replicates, all at $H = 5$. The validation sequence found an open

(bright) channel in 539 images and an occluded (dark) channel in 310, and it reported no channel in 141. With every diagnostic figure enabled and PyTorch running on the CPU, the run took about 80 s per image, as estimated from the timestamps of its output files. The closure run processed 123 open/closed pairs across four series in about 5 s per pair with SAM output cached. Of these pairs, 99 passed every gate: 63 with an accepted arc fit, 3 with the parabola fallback, and 33 as validated zero closure. The gate discarded the other 24 pairs, 22 of them on fit quality, 1 as photometrically inconsistent, and 1 because SAM failed on the open image. Every excluded pair therefore carries a machine-readable reason and, where recoverable, the exact redo filename on the re-imaging list. The top-view program processed its 112 images in 4 to 9 s each. In all three cases the important measure is where the expert's time goes. It goes into reviewing diagnostic figures and settling the re-imaging list, not into drawing masks or transcribing measurements.

3.6. One Program or Three? Architectural Assessment

The two cross-section programs share the SAM service, the validation sequence, filename parsing, caching, diagnostic figures, and replicate statistics. That overlap makes it natural to ask whether they should be one general program. After maintaining both, we think they should remain two thin applications over one shared core. The reason lies in their measurement semantics rather than in their code. The printability program answers a categorical question about one image (did a channel form, and in what state?). Its unit of work is the image, its output is a labeled map, and its statistics are votes. The closure program answers a continuous question about a pair of images (how far did the membrane move between states?). Its unit of work is the registered pair, its output is a fitted physical model, and its statistics are gated means. Forcing both through one interface would hide the differences that matter, such as registration, differential photometry, model fitting, and the distinction between discarding and zeroing, under configuration flags. The genuinely shared layers are already shared in practice, since the closure program imports the validation sequence from the printability code, and they are natural candidates for a formal common library.

The top-view program answers a third kind of question, about a one-dimensional structure (where along the channel does it narrow or close?). That program does not use SAM. A top-view channel appears as a pair of thin dark lines, often faint on one side, rather than as a bright region. Line tracking by dynamic programming, anchored at the ports and constrained by the design width, uses the geometry that the printer imposes: channels are parallel, evenly pitched, and run from face to face. The three programs do share principles, namely filename-encoded metadata, dimensions normalized within the image, per-image diagnostic overlays, and tables that keep the underlying numbers next to the classes. A shared core with one entry point per question is the organization we recommend to groups adapting this code, and it is the direction we are pursuing.

3.7. Limitations

The cross-section method requires optical access by design. An experimenter must build each test device as a cross-sectional mimic with open ends, so the method characterizes representative test devices rather than arbitrary production devices, which is the same trade the companion framework makes deliberately⁶. Each cross-section view measures one section plane, and the top-view program now covers variation along the channel, but only for the printability devices. The intensity-loss measurement of closure is comparative and dimensionless by construction. It cannot report contact pressure or seal quality, which the companion study measures by flow testing⁶. Physical calibration of the cross-section views relies on a known scale-bar value, because OCR of the footer labels proved unreliable, and other microscopes would need their own solution. Opaque or strongly scattering resins would reduce the contrast on which both imaging modes rest,

although the clear resins common in 3D-printed microfluidics are the favorable case. We chose the parameters of the validation sequence against expert review, but they fit this combination of microscope, resin, and geometry. A new imaging context should therefore be expected to need retuning.

The top-view program is at an earlier stage than the cross-section programs, and its limitations are correspondingly larger. Its thresholds were set on the same 14 expert-reviewed images used to validate it, so the agreement reported in Section 3.4 is an optimistic estimate. When one wall is faint, the tracks lose it, and the channel becomes unresolved or a low-confidence call. When a channel ends in a rounded blind tip without a dark rim, the program can bridge the tip as a dropout and report the channel as formed or necked. We observed this error on several $H = 5$, $W = 70$ devices. The shares of formed channels at such widths are therefore overestimated. Touchdown detection needs a closed inner outline, so the program misses a faint or broken one. The same detector can also count an enclosed air bubble as touchdown. At the narrowest designs, below about 12 image pixels, the channel count relies on the design pitch because the ports are tiny wedges. Devices without dark port edges come out with low confidence, and the design summaries leave them out. We report the top-view results as a first application and will update them as the program matures.

4. Conclusions

We have described imaging methods and three open-source analysis programs that turn the internal state of monolithically 3D-printed microfluidic devices into automated, auditable measurements, using a zoom stereomicroscope with transmitted illumination and purpose-designed test devices. The cross-section programs segment with a zero-shot model and apply all domain knowledge as transparent classical tests, and they discard untrustworthy measurements instead of recording them as zero. In the companion study these programs produced the printability maps, sag measurements, closure curves, and membrane deflection coefficients on which the valve-design framework rests⁶. The new top-view program extends the same approach along the channel, where it finds necks and closures that a cross-section at the channel end cannot see. We release all three programs with their diagnostic tooling and parameter documentation so that other groups can apply them to their own printers, resins, and devices.

Code Availability

The source code is available under the MIT license at https://github.com/iomega15/SAG_ANALYSIS_SAM_SEGMENTATION (sag/printability program; MATLAB R2026a with the Image Processing and Computer Vision Toolboxes, plus the Python SAM service), https://github.com/iomega15/VALVE_CLOSURE_ANALYSIS (valve-closure program; MATLAB R2026a, additionally the Optimization Toolbox), and https://github.com/iomega15/TOP_VIEW_CHANNEL_PINCH (top-view program; Python 3.10). Each repository documents its environment, parameters, and diagnostic outputs. The SAM ViT-B checkpoint is available from its original release¹⁰. The top-view program is under active development, and this preprint describes the version of 22 September 2026.

CRedit authorship contribution statement

Roman Voronov: Conceptualization, Supervision, Project administration, Funding acquisition, Software (MATLAB/Python image-analysis programs), Formal analysis, Writing – original draft, Writing – review & editing. **Huu Anh Minh Nguyen:** Investigation (device fabrication and microscopy), Data curation, Validation (expert review of program outputs), Writing – review & editing. **Nhu Doan Quynh Nguyen:** Investigation (device fabrication and microscopy), Data curation. **Quynh Ai Vuong:** Investigation (device fabrication, cross-section and top-view microscopy), Data curation. **Mark Volosov:** Resources (custom actuation/control hardware), Software. **Dae Jung Martins Cruz:** Resources (custom actuation/control hardware), Software. **Bernardino Almanzar:** Resources (custom

actuation/control hardware), Software. **Abigail Goz:** Software (OpenSCAD device generation). **Maanit Khanna:** Software (OpenSCAD device generation). **Rishabh Jitender:** Software (OpenSCAD device generation). **Sergei Chapek:** Conceptualization (3D device design). **Aidan Kinney:** Investigation (microfluidic experiments).

Acknowledgments

Funding: This work was supported by the National Institute of General Medical Sciences of the National Institutes of Health (NIH) under award number 1R15GM145610-01. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. Additional support came from the National Science Foundation (NSF) Award Number 2141029.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used OpenAI ChatGPT, Google Gemini Pro, and Anthropic Claude (Opus and Fable) to check grammar, to assist with the development of the image-analysis and data-processing code described herein, and to assist in drafting the manuscript from the source code and its documentation. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Declaration of Competing Interests

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

References

- (1) Bhattacharjee, N.; Urrios, A.; Kang, S.; Folch, A. The upcoming 3D-printing revolution in microfluidics. *Lab on a Chip* **2016**, *16* (10), 1720-1742. DOI: 10.1039/c6lc00163g.
- (2) Au, A. K.; Huynh, W.; Horowitz, L. F.; Folch, A. 3D-printed microfluidics. *Angewandte Chemie International Edition* **2016**, *55* (12), 3862-3881. DOI: 10.1002/anie.201504382.
- (3) Gong, H.; Woolley, A. T.; Nordin, G. P. High density 3D printed microfluidic valves, pumps, and multiplexers. *Lab on a Chip* **2016**, *16* (13), 2450-2458. DOI: 10.1039/c6lc00565a.
- (4) Sadyrin, E. V.; Nikolaev, A. L.; Chapek, S. V.; Nazarenko, D. V.; Aizikovich, S. M.; Wang, Y.-C. Manufacturing quality evaluation of photopolymer resin 3D-printed scaffolds using microtomography. In *Sixty Shades of Generalized Continua*, Advanced Structured Materials, Springer, 2023; pp 619-630.
- (5) Sadyrin, E. V.; Karotkiyan, R. V.; Nikolaev, A. L.; Chapek, S. V.; Lapina, P. A.; Wang, Y.-C. In situ mechanical testing and microtomography of 3D-printed hard tissue scaffolds for bioengineering applications. In *Dynamics of Discrete and Continuum Structures and Media*, Advanced Structured Materials, Springer, 2025; pp 599-616.
- (6) Nguyen, H. A. M.; Vuong, Q. A.; Nguyen, N. D. Q.; Martins Cruz, D. J.; Almanzar, B.; Kinney, A.; Goz, A.; Khanna, M.; Jitender, R.; Chapek, S.; et al. A geometry-first design framework and interactive design tool for 3D-printed microfluidic valves using new photopolymer resins. *engrXiv* 2026. DOI: 10.31224/8053.
- (7) Lee, Y.-S.; Bhattacharjee, N.; Folch, A. 3D-printed Quake-style microvalves and micropumps. *Lab on a Chip* **2018**, *18* (8), 1207-1214. DOI: 10.1039/c8lc00001h.
- (8) Rogers, C. I.; Qaderi, K.; Woolley, A. T.; Nordin, G. P. 3D printed microfluidic devices with integrated valves. *Biomicrofluidics* **2015**, *9* (1), 016501. DOI: 10.1063/1.4905840.

- (9) Hinnen, H.; Viglione, M.; Munro, T. R.; Woolley, A. T.; Nordin, G. P. 3D-Printed Microfluidic One-Way Valves and Pumps. *Micromachines* **2023**, *14* (7), 1286. DOI: 10.3390/mi14071286. 777
778
- (10) Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A. C.; Lo, W.-Y.; et al. Segment anything. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023; IEEE: pp 3992-4003. DOI: 10.1109/iccv51070.2023.00371. 779
780
781
- (11) Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015)*, Lecture Notes in Computer Science, Springer, 2015; pp 234-241. 782
783
784
- (12) Otsu, N. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics* **1979**, *9* (1), 62-66. DOI: 10.1109/tsmc.1979.4310076. 785
786
- (13) Canny, J. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **1986**, *PAMI-8* (6), 679-698. DOI: 10.1109/tpami.1986.4767851. 787
788
- (14) Leong, K. M.; Sun, A. Y.; Quach, M. L.; Lin, C. H.; Craig, C. A.; Guo, F.; Robinson, T. R.; Chang, M. M.; Olanrewaju, A. O. Democratizing access to microfluidics: rapid prototyping of open microchannels with low-cost LCD 3D printers. *ACS Omega* **2024**, *9* (45), 45537-45544. DOI: 789
790
791
792
793