Supplementary information

S1 System Specifications

More detailed information on system performance and the methods used to measure key metrics are summarized here.

S1.1 Magnification and Field of View

The theoretical magnification for the FiLM-Scope is given by:

$$M = f/F = 14.64\,mm/100\,mm = 0.1464 \tag{S1}$$

where f is the focal length of the array lenses and F is the focal length of the primary lens.

The true magnification value is less than this theoretical value, because the lenses were placed slightly outside of a true 4-f configuration in order to increase the working distance of the system. Additionally, the magnification varies slightly between individual cameras, because each array lens is focused individually. Magnification for each lens was measured by acquiring an image of a graph target at the most in-focus plane for the central camera, then finding the distance in pixels between the graph vertices. The magnification reported here is an average over the field-of-view for each camera.

The magnification varied between 0.1205 and 0.1220, with a **mean value of 0.1212** and a standard deviation of 0.0004. Given our sensor size of 4096 x 3120 pixels, and pixel pitch of 1.1 μm, this corresponds to an **average field-of-view of 37.2 x 28.3** mm.

S1.2 Lateral Resolution

Using $\lambda = 530\ nm$ for wavelength of light, and a theoretical $NA = \tan^{-1}(\frac{d/2}{F}) = 0.028$ ($d = 5.7\ mm$ is the diameter of the array lenses, and $F = 100\ mm$ is the focal length of the primary lens), the theoretical diffraction limited resolution of the system is given by:

$$\phi_{lat,diff} = \lambda/(2 \cdot NA) = 9.3\ \mu m \tag{S2}$$

To measure the on-axis lateral resolution in each camera, we imaged a resolution target at 13 distances from the primary lens along the optical axis of the system, over a range of 6 mm. At each plane, we took the full-width at half maximum (FWHM) of the line-spread function (LSF) for both a vertical and horizontal line, and repeated for all 48 cameras (Figure S1b). We then found the minimum FWHM for each camera (Figure S1c).

The results are summarized in Figure S1d. The resolution for a camera is related to its position in the camera array, with cameras in the middle of the array exhibiting the finest resolution. For many cameras, resolution differed between the x and y dimensions. For instance, cameras in the center left of the array had fine resolution in the x (vertical) dimension, and worse resolution in the y (horizontal) dimension. Cameras in the top center of the array had fine resolution in the y (horizontal) dimension, and poor resolution in the x (vertical) dimension. The resolution averaged between both dimensions varied between 19 μm and 30.4 μm, with an **average value of 23.43 μm and a standard deviation of 2.57 μm**. These results suggest that the performance of the FiLM-Scope could be improved by re-designing the primary lens to achieve more uniform resolution among the cameras.

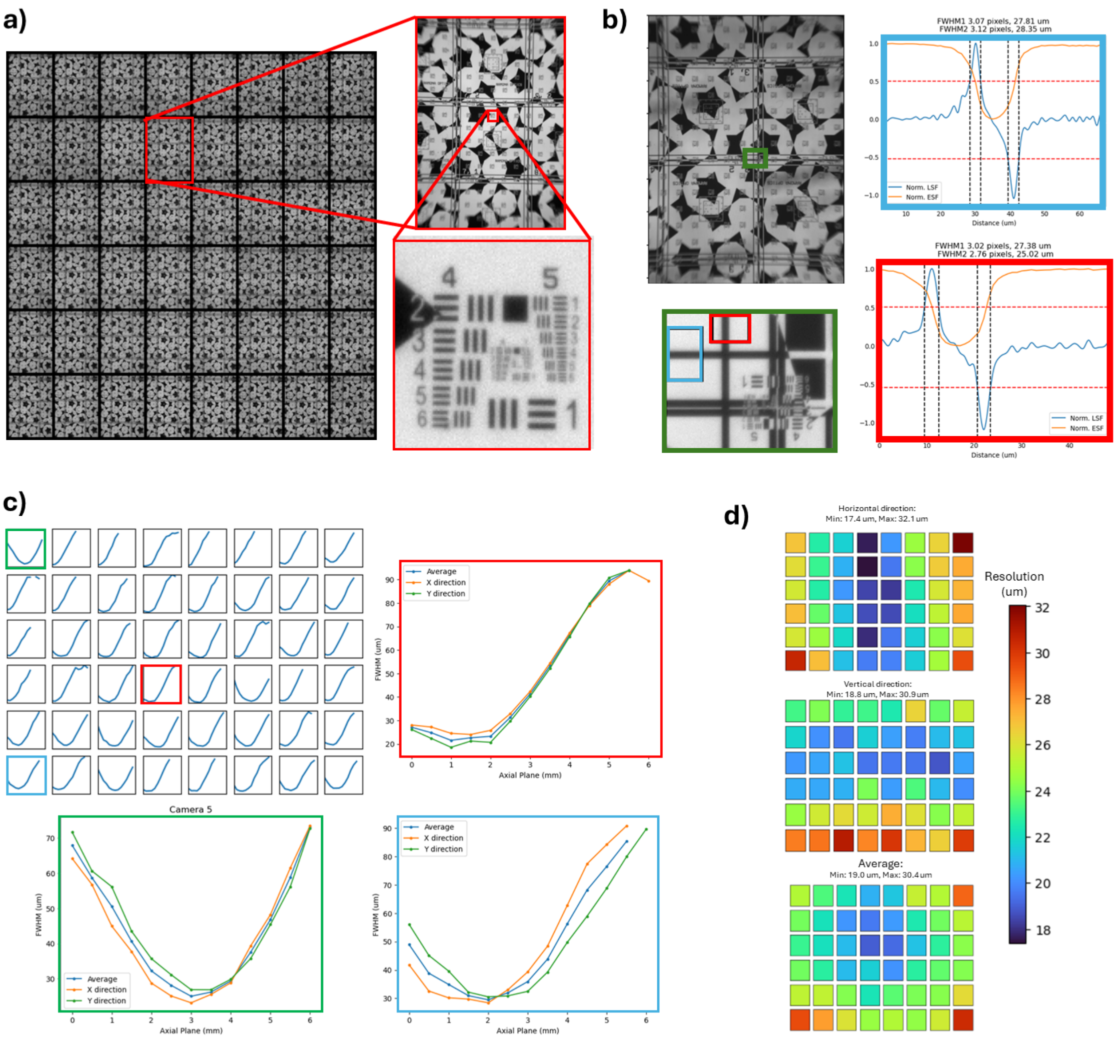

Fig S1 Lateral resolution plots. **a)** Snapshot of USAF resolution target. On-axis, we are able to resolve down to group 5, element 5 in the highlighted camera. **b)** Left: Example image of the horizontal and vertical lines used to measure FWHM resolution of system. Right: Plotted edge-spread and line-spread functions from the blue and red insets of the left image. This was repeated for all 48 cameras at 13 axial planes. The vertical and horizontal FWHM values were averaged to create the plot in c). **c)** Average FWHM plots for all 48 cameras. The red, blue, and green insets highlight these plots for 3 cameras. From these plots, we can see there is sometimes a difference between the vertical and horizontal resolution. Additionally, the location of the optimal focal plane varies between cameras. **d)** On-axis resolution values for the 48 cameras. Top: FWHM values in the horizontal direction. Middle: FWHM values in the vertical direction. Bottom: average of vertical and horizontal FWHM values. From this, we can see that resolution is dependent on a camera's location in the array.

S1.3 Axial Resolution

To quantify axial resolution, we use the definition from Guo (2019),[22] which is the minimum axial distance between two laterally aligned points to resolve them as separate points in a given camera.

Each camera has its own axial resolution, and the axial resolution for the system can be reported as the best resolution amongst the cameras.

For a given camera i, the theoretical axial resolution is given by:

$$\phi_{ax,i} = \phi_{lat,i} / \tan(\theta_i) \tag{S3}$$

where ϕ_{lat} is the lateral resolution and θ_i is the angle between the system optical axis and the chief ray for camera i (See Figure 2a, in the main text).

To estimate the axial resolution for each camera, we first found θ_i in both the x and y dimensions for each camera by using the calibration results, $S_i(p)$ (see Methods section). If p_x and p_y are here the central pixel values for camera i, M_i is the magnification, and $\rho = 1.1\ \mu m$ is the pixel pitch, we have:

$$\theta_i = \tan^{-1}(S_i(p_x, p_y) \cdot \rho / M_i) \tag{S4}$$

Figure S2a, shows these values in the x and y directions, as well as their magnitudes.

From there, we can find the geometric and diffraction limited axial resolution using the lateral resolution values from S1.2. The axial resolution values for all cameras are shown in Figure S2b - c. **We calculated the geometric resolution as 49 $\mu\mathrm{m}$ and the diffraction limited resolution as 83 $\mu\mathrm{m}$.** The large discrepancy between the geometric and diffraction limited resolutions is due to the poor lateral resolution in the outer-most cameras of the array. These cameras have the largest values for θ_i and can thus theoretically provide the best axial resolution, but this is not fully realized due to their poor lateral resolution.

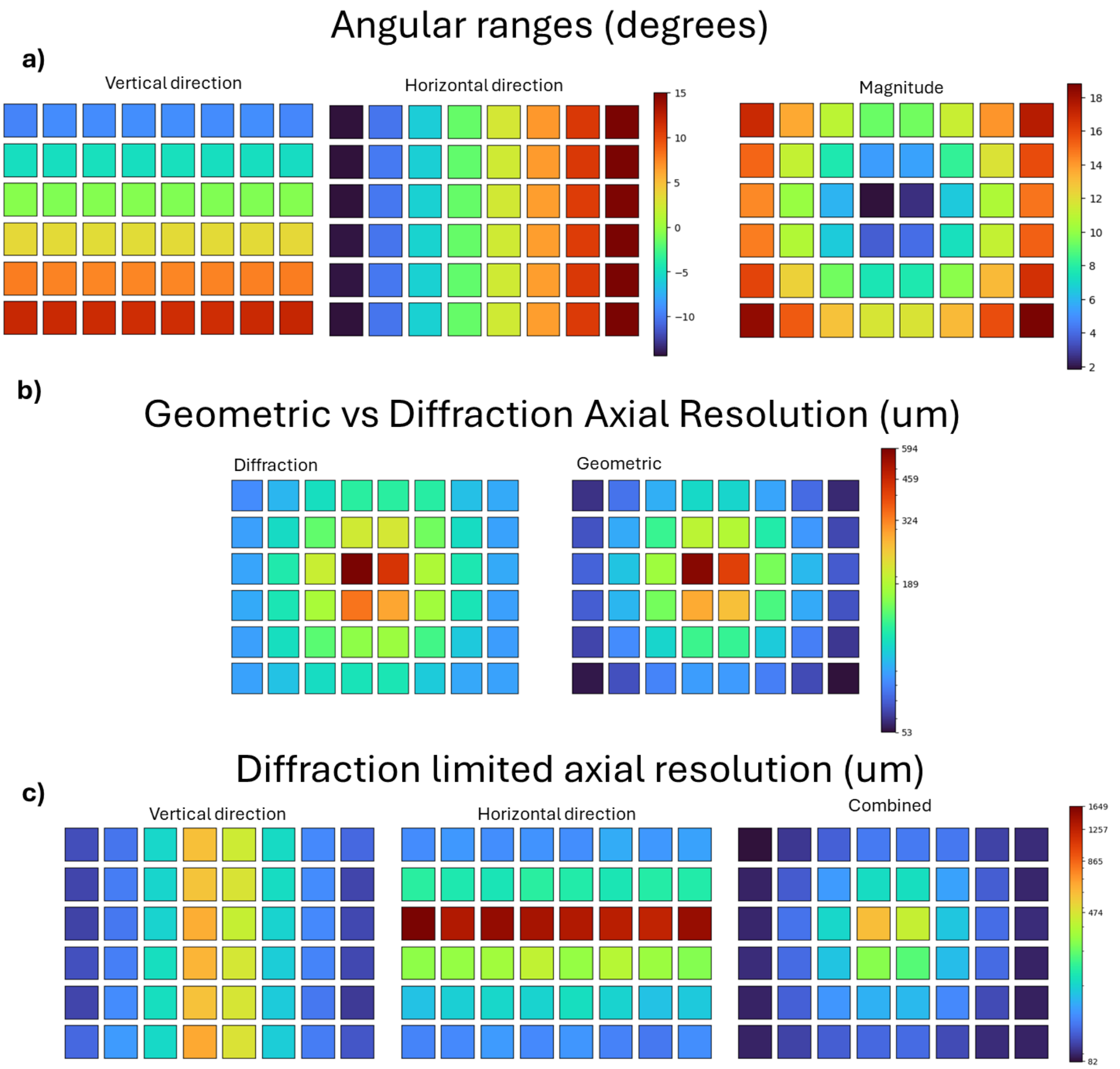

Fig S2 Axial Resolution a) Center angle of acceptance, θ_i, for the 48 cameras. Left: vertical direction, middle: horizontal direction, right: magnitude. **b)** Comparison of geometric and diffraction limited axial resolution. For the center cameras, the two values are similar. However, for the edge cameras, the diffraction limited resolution is noticeably worse than the geometric limited resolution. This is due to the worsening lateral resolution in the cameras at the edge of the array. **c)** Diffraction limited axial resolution for the 48 cameras.

56 ### *S1.4 Depth of Field*

57 The theoretical depth-of-field (DOF) for the FiLM-Scope can be given by:

$$DOF_{theor} = \frac{(n^2 - NA^2)}{NA^2} \cdot \lambda = \frac{1 - 0.028^2}{0.028^2} \cdot 530\ nm = 675\ \mu m \tag{S5}$$

58 To measure the DOF, we used the lateral FWHM resolution values computed in S1.2. We then

fit a normal curve to the 1/FWHM values (Figure S3a) and identified the DOF from that curve. Figure S3b shows those values for 46 of the 48 cameras (for the remaining two, the normal curve could not be accurately fit, so those values are left blank). We consider two values for DOF: the estimated DOF from the fit curve, as well as the "usable" DOF. Because the working distance of the FiLM-Scope is very short, the DOF for some cameras extends inside the glass surface (for instance, camera 20 in Figure S3). Thus, the DOF that can be practically used is more limited. We refer to the portion of the DOF outside the glass surface of the lens as the "usable DOF".

The values of the full DOF varied bewteen 2.65 mm and 5.65 mm, with a mean value of 3.68 mm and standard deviation of 0.61 mm. **The usable DOF varied between 1.37 $\mathbf{mm}$ and 4.45 $\mathbf{mm}$**, with a mean value of 3.13 mm and standard deviation of 0.85 mm.

It is worth noting that while the resolution is be worse outside the reported DOF, we are still able to achieve reasonable 3D results well outside this range. We show accurate results up to about 1 cm depth ranges.

S1.5 Aberrations

The effective FOV of the FiLM-Scope is limited by off-axis coma aberrations, which we begin to characterize in this section. To measure the aberrations, we imaged a square within a USAF resolution target at 25 locations across the FiLM-Scope's FOV, at the optimal on-axis focal plane for camera 20 (which is one of the four central cameras in the array). For all 48 cameras, we found the line-spread function (LSF) across all four sides of the squares in the 25 positions, and used these to characterize vertical and horizontal resolution.

The results for camera 20 are shown in Figure S4. Figure S4a shows the images of the square target from the 25 positions within the FOV of camera 20, and the line-spread functions for the

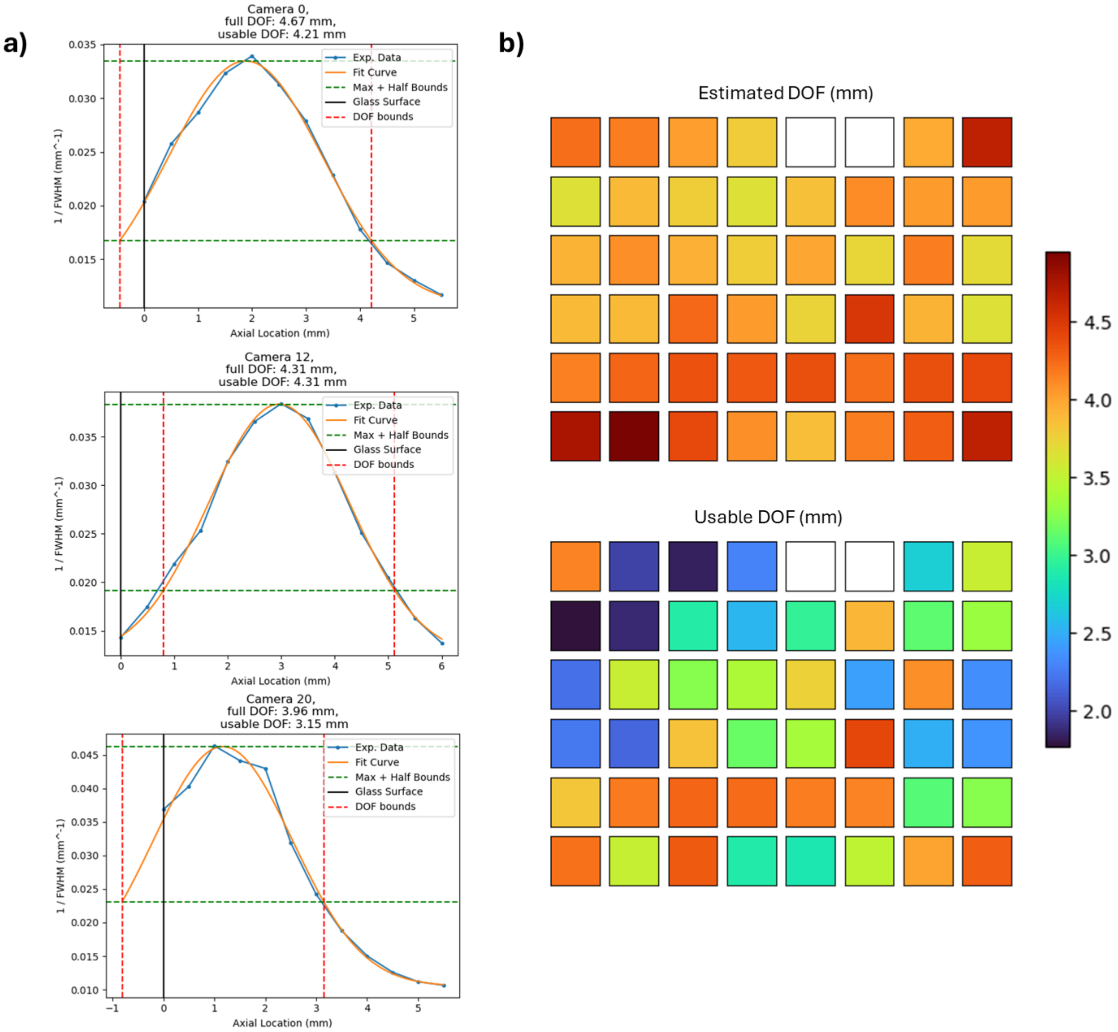

Fig S3 Depth-of-field (DOF). DOF for the FiLM-Scope was estimated by first talking FWHM resolution measurements, ϕ, at 13 axial planes for each of the 48 cameras, then fitting a normal curve to $1/\phi$. For many cameras, a portion of the estimated DOF lies inside the glass of the primary lens, so the usable DOF may be smaller than the width of the curve. **a)** Curves to estimate DOF from experimental FWHM measurements for cameras 0, 12, and 20. For camera 12, the entire DOF lies past the glass surface of the lens (black line), so the usable DOF is the same as the estimated DOF. For cameras 0 and 20, a portion of the estimated DOF lies inside the glass surface of the lens, so the usable DOF is smaller than the estimated DOF. **b)** Estimated (top) and usable (bottom) DOF for each of the 48 cameras. For the two white squares, the DOF could not be accurately estimated from the available data for those cameras, so they are left blank.

edges of these squares are shown in Figure S4b. While the edges in the center of the FOV have relatively symmetric LSFs (green inset), away from the center of the FOV the LSFs become asymmetric and vary based on the square's location (red inset). To better visualize this, we found the full-width at half max (FWHM) value for each LSF, and further split this into top/bottom or left/right half-width values by finding the peak point in the LSF (see red and green insights in Figure S4b). After finding these values at each of the 25 positions, we fit four 2D polynomials to give the left, right, top and bottom values over the full FOV. Those polynomials are shown in Figure S4c. From these plots, we can see that the system exhibits strong positive coma aberrations.

In Figure S5, we show the resolution across the FOV of all 48 cameras. These values were found by averaging the FWHM of the LSF in the horizontal and vertical directions, and fitting the results across the FOV. From these plots, we can see that the resolution drops significantly away from the center of the FOV, suggesting that the system performance could be improved in the future by using custom designed lenses better optimized for this purpose.

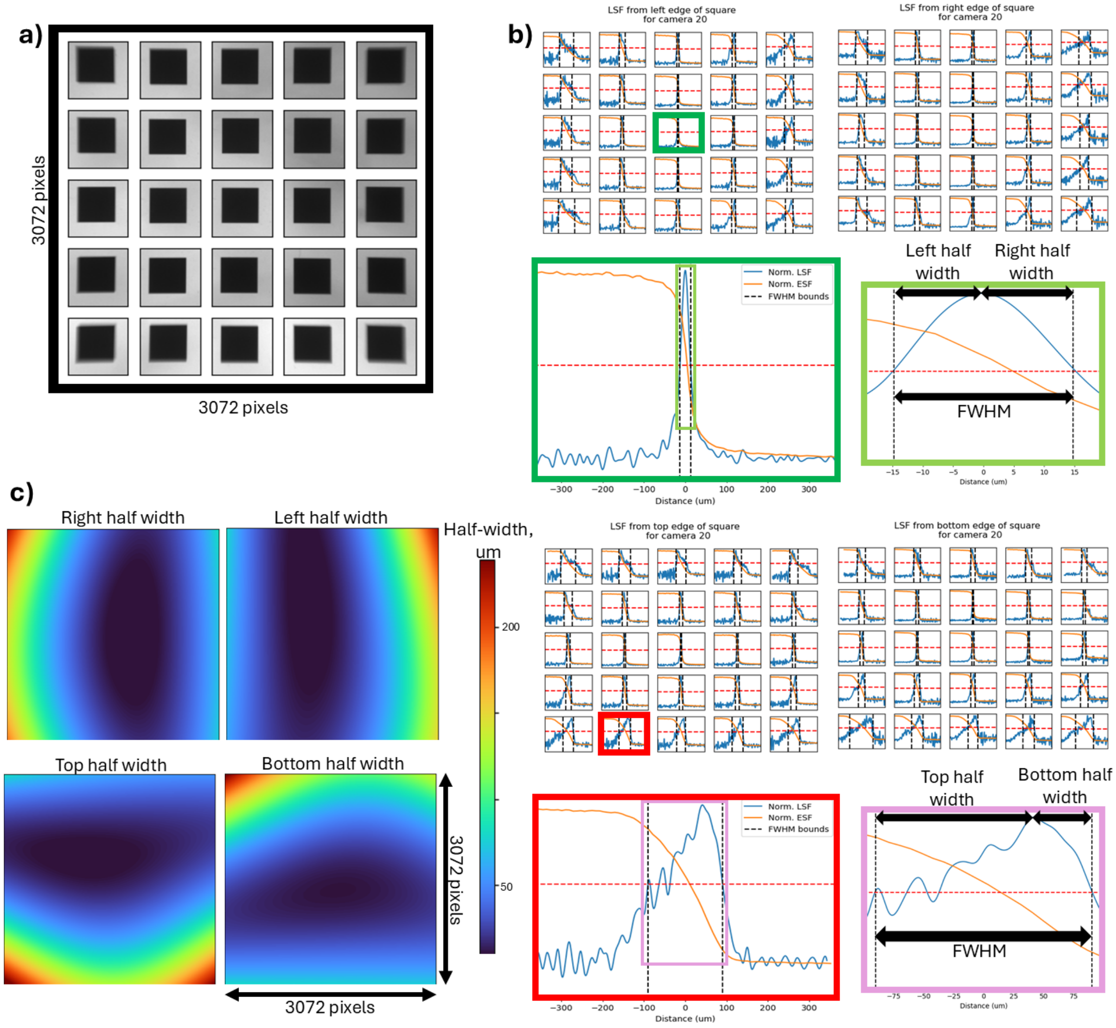

Fig S4 Aberrations for reference camera. These plots show how aberrations were measured in a single camera. This process was repeated for all 48 cameras. In all cameras, there are strong positive coma aberrations. **a)** To measure resolution across the FOV, we translated a square target to 25 positions within the FOV of the camera, at the optimal focal plane for the reference camera. **b)** We then computed the edge spread function (ESF) and line spread function (LSF) across all four edges of the square, for each of the 25 positions. Top left: LSF/ESF plots for left edge of the square, top right: LSF/ESF plots for right edge of the square, bottom left: LSF/ESF plots for top edge of the square, bottom right: LSF/ESF plots for bottom edge of the square. From these, we can see that the LSF is quite asymmetrical, depending on the square's position within the FOV. In the green inset, for the center of the FOV, the LSF is symmetrical, and the left and right half-widths are roughly the same width. In the red inset, for a square at the bottom edge of the FOV, the LSF is unbalanced: the top half is considerably longer than the bottom half. **c)** These plots were generated by taking the left, right, top, and bottom half-FWHM values for each of the 25 positions, and fitting a polynomial to visualize those values over the full FOV. The system's positive coma aberration is apparent. These suggest that the point-spread-function will have a tail pointing towards the center of the FOV.

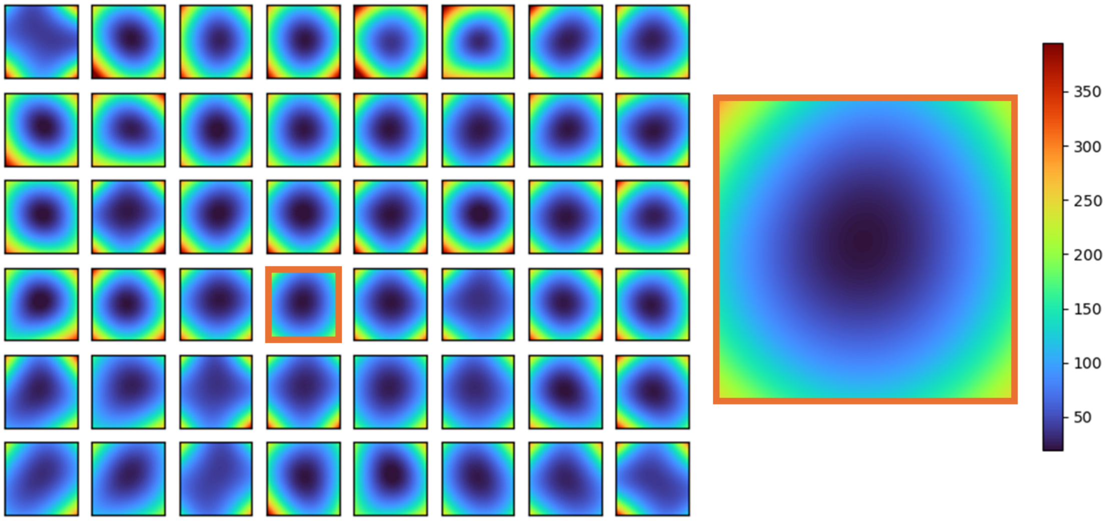

Fig S5 Aberrations for all cameras. Here, we summarize the aberrations across the FOVs of all 48 cameras. These were computed by averaging the vertical and horizontal FWHM values computed as shown in Figure S4 for camera 20, and fitting a polynomial to describe the resolution values across the full FOV. There is significant resolution fall-off towards the edge of the FOV for all the cameras. This could be improved in the future through the use of a more optimized or custom-designed primary lens. Scale bar is in millimeters.

S2 System Calibration

The flow diagram for system calibration is shown in Figure S6a. The process consists of five steps, which are briefly described here. The full calibration code with example datasets is available on the GitHub page.

Step 1: Acquire Graph Images: To perform calibration, we acquire images of a piece of graph paper, which is held flat against a sheet of glass. The graph is translated axially away from the primary lens through the DOF of the system, in steps of 1 mm, while acquiring a snapshot image at each step. Example images are shown in Figure S6b.

Step 2: Extract Vertex Locations: Next, we extract all the vertex locations from each of the graph images. This is currently done with a custom script, which can successfully localize both in-focus and aberrated vertices. The calibration is most successful when the majority of vertices are identified, but the procedure has enough redundancy that the results are adequate even when some vertices are missed.

Step 3: Intra-camera alignment: The vertex locations extracted in Step 2 must be matched between planes to fit polynomial coefficients for $S_{i,x}(p)$ and $S_{i,y}(p)$ (as described in the Methods section of the main text). To match vertices, we assume the spacing between different vertices is much larger than the amount a single vertex will shift between images at adjacent planes. After matching the vertices between planes, we fit two lines, with slopes m_x and m_y in units of pixels/plane, to describe how far the image of the vertex shifted in x and y when the graph was translated axially between planes. The slopes are then converted to pixels/mm, using the known translation of the graph between snapshots in Step 1.

Finally, we can fit coefficients for the polynomial equations $S_{i,x}(p)$ and $S_{i,y}(p)$ (see Equations 8, 9). This is done via least squares fit, where the vertex locations on the reference plane are taken as the independent variable, and the slopes m_x and m_y are the dependent variables. The red and blue insets from Figure S6b show an example outcome of this step: the extracted vertex locations are plotted in red, the fit lines with slopes m_x and m_y are shown in orange, and the outputs of the equations $S_{i,x}(p)$ and $S_{i,y}(p)$ are shown in blue. From this plot, we can see strong agreement between the orange and blue lines, suggesting that our calibration approach is successfully capturing the physics of the FiLM-Scope.

Step 4: Inter-camera alignment Using the same vertices extracted in Step 2, we can then calibrate for the FOV shift between cameras. Note that in an ideal Fourier Light Field system, there would be no FOV shift at the object plane. However, to account for misalignment in this system, it is important to complete this calibration step. We begin by selecting the snapshot acquired closest to the object plane of the system (typically the plane most in focus for the central array camera), then matching the vertices between the 48 images. This is done by first manually selecting a single matching point in all 48 images, then using that approximate FOV shift to find the precise shift between vertices.

Similar to step 2, we perform a least squares fit to find the polynomial coefficients for the equations $O_{i,x}(p)$ and $O_{i,y}(p)$ (Equations 10, 11). The independent variable is the vertex location in the image from camera i, and the dependent variable is the pixel shift between the vertex location in camera i and its location in the reference camera. An example outcome of this calibration step is shown in Figure S6c. The left graph shows the FOV shift between the center of each camera and the reference camera, while the red, blue, and green insets highlight how the shift varies across the

FOV of a camera.

Step 5: Generate Dense Mappings After steps 1-4 are complete, the information from intra- and inter-camera calibration can be used to compute dense mappings during reconstruction.

For each camera, we compute maps to give both the "shift slopes" encoded by S_x and S_y and the "inter-camera shifts" encoded by O_x and O_y. We will refer to the shift slope maps as M_s, and the inter-camera maps as M_c. These maps are formed by computing the values for $S_i(p)$ and $O_i(p)$ for each pixel location $p = (p_x, p_y)$. The outcome of this step is 48 x 4 maps: $M_{c,i,x}$, $M_{c,i,y}$, $M_{s,i,x}$, $M_{s,i,y}$ for each camera i.

Because of how back projection is performed in our algorithm using Pytorch's *torch.nn.functional.grid_sample* function (see Appendix C and Equation 19 in the main text), these maps must be warped to the perspective of the reference camera, to give the values for $\hat{S}(p)$ and $\hat{O}(p)$. Each of the four warped maps $\hat{M}$ ($\hat{M}_{c,i,x}$, $\hat{M}_{c,i,y}$, $\hat{M}_{s,i,x}$, or $\hat{M}_{s,i,y}$) is computed by warping the corresponding map M with *grid_sample* using M_c.

At the end of calibration, all key information is saved in a single calibration file, which is stored alongside the acquired image datasets. Along with the calibration results, we store parameters including the pixel pitch and size of the graph target used in calibration, which can be used to calculate magnification.

S3 Reconstruction Implementation Details

The full calibration and reconstruction code is available on GitHub, alongside example datasets. Here, we highlight a few key features of the pipeline that were not described in the main text.

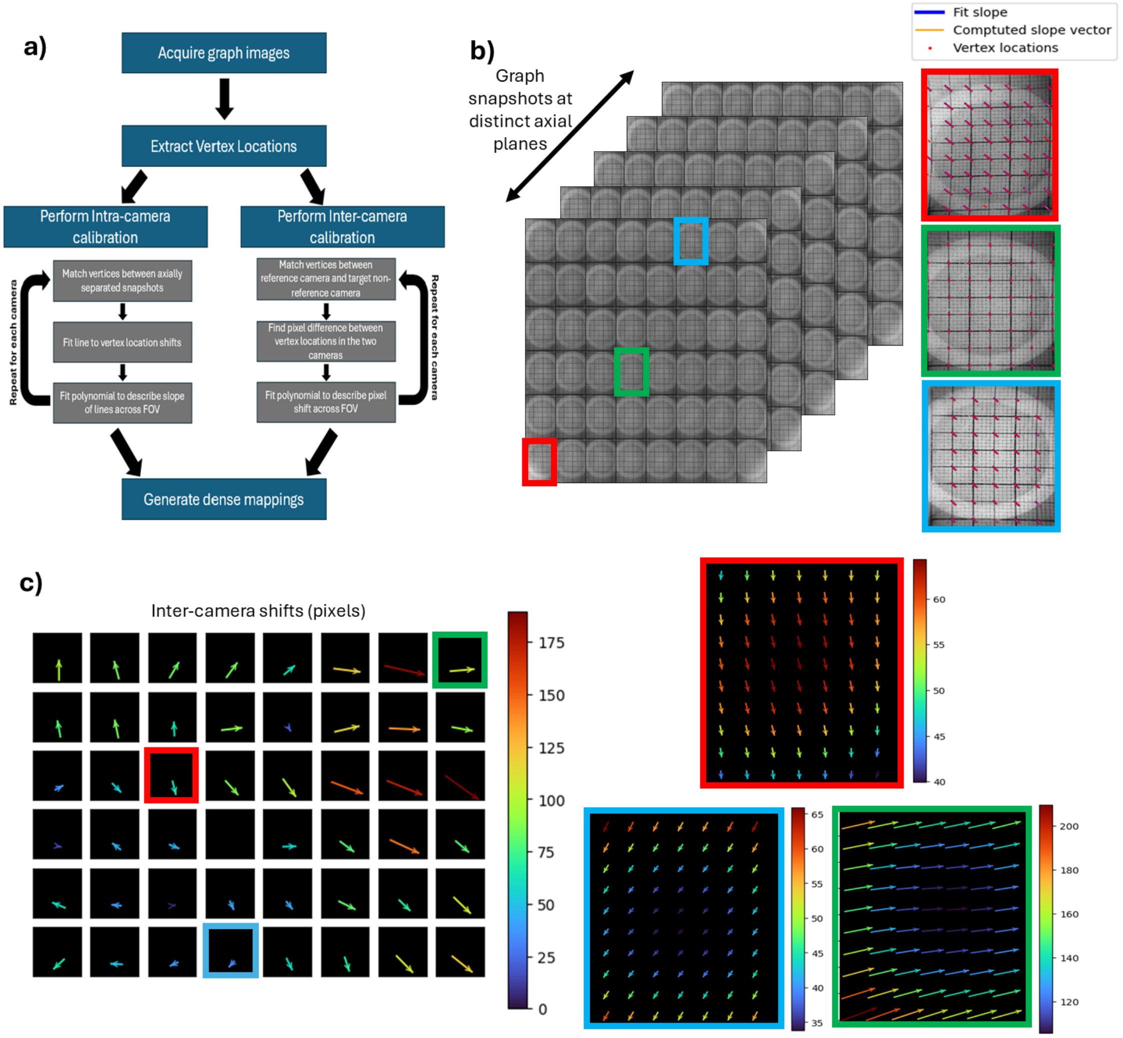
a)
Acquire graph images
Extract Vertex Locations
Perform Intra-camera calibration
Perform Inter-camera calibration
Match vertices between axially separated snapshots
Fit line to vertex location shifts
Fit polynomial to describe slope of lines across FOV
Repeat for each camera
Match vertices between reference camera and target non-reference camera
Find pixel difference between vertex locations in the two cameras
Fit polynomial to describe pixel shift across FOV
Repeat for each camera
Generate dense mappings
b)
Graph snapshots at distinct axial planes
Fit slope
Comptuted slope vector
Vertex locations
c)
Inter-camera shifts (pixels)
175
150
125
100
75
50
25
0
60
55
50
45
40
65
60
55
50
45
40
35
200
180
160
140
120

Fig S6 Calibration. **a)** Flow-chart for the calibration procedure, including both intra-camera calibration to find coefficients for the "shift ratios" $S_{i,x}(p)$ and $S_{i,y}(p)$, and inter-camera calibration to find coefficients for the "inter-camera shifts" $O_{i,x}(p)$ and $O_{i,y}(p)$. **b)** Left: example of 5 snapshots of graph target acquired at distinct axial planes, spaced 1 mm apart. Right: zoom-in on three individual images of the graph. In each, the locations of vertices from the snapshots at different planes are shown in red. The line fit between the shifted locations for a single vertex is shown in yellow, and the output of S_i with the fit polynomial coefficients is shown in blue. **c)** Results of inter-camera calibration (intra-camera calibration is shown in the main text, Figure 2). Left: vectors showing average value of O_i for each camera. Right: For three cameras, plots of how $O_i(p)$ changes across the camera's FOV.

S3.1 Key arguments

To ensure optimal reconstrucion results and GPU utilization, a number of arguments can be specified by the user in the configuration file. While most of these were held constant for samples shown in this work, we highlight a few of the arguments that are most frequently modified between samples.

Height range: Prior to the start of reconstruction, the user must specify an approximate height range for the object. This will determine the depth through which the initial back-projection volume is formed (see main text Figure 2). Currently, this height range is selected manually. This is done by forming a digital z-stack of the image, and identifying the height range over which features are in focus. In future iterations, this could be automated by using a focus metric to roughly identify the top and bottom extents of the object from the digital z-stack.

Number of Planes in Volume: The spacing between planes in the back-projection volume is key in determining the precision of the reconstruction, and should generally be matched to the expected axial resolution. However, it often makes sense to use a smaller number of planes to save GPU space, particularly when reconstructing over a large depth. The number of planes is specified by the user in the configuration file, and should be a multiple of 32 when using a U-Net with 4 max-pool layers.

Crop information: In many instances, image reconstruction is performed in cropped segments, either due to limited space on the GPU, or because the sample does not fill the field-of-view of the system. Cropping is handled by the algorithm, to ensure the calibration maps are properly matched to the cropped section of the image. The user can manually specify cropping information as an argument. The crop center location, width, and height are all specified in normalized units. When the dataset is loaded, these are converted to pixels, based on the down-sampling level being used, and adjusted to ensure compatibility with the U-Net architecture.

Estimated height: Particularly when working with cropped images, providing an estimated height can ensure more complete reconstruction results. Because the image of an object shifts based on its height, if the same crop location is used in all 48 images, there may be limited overlap between their respective FOVs. This can be compensated for by providing an approximate estimated height, which is used to select the cropped regions for each image. Currently, this value defaults to the center of the provided height range, but it can also be manually specified in the configuration file.

Downsample amount: While the most precise reconstructions are generally achieved by performing reconstruction with full resolution images, the reconstruction time can be sped up considerably by downsampling the images. Additionally, to increase frame rate, videos may be acquired with pixel binning. For many samples, this may only marginally reduce the precision of the reconstruction results. The desired downsample amount can be specified by the user in the configuration file, and the calibration information will be appropriately matched when the dataset is loaded.

U-Net architecture: The main network in our code is a 3D U-Net. The architecture of this U-Net can be adjusted (including the number of layers, the channels per layer, and the stride-length in all 3 dimensions). For most samples, to achieve the best results, we used a network with 4 downsampling steps, with 2 x 2 x 2 stride length. We used 8 convolution channels in the first layer, and 16 in all subsequent layers. While this was considerably fewer channels than in the original MVS-Net,[33–35] we found that this allowed us to save GPU space without reducing reconstruction quality.

Rectification: By default, the depth maps produced by this algorithm are aligned with the image from the reference camera/viewpoint. However, the maps can also be produced to align with a rectified viewpoint along the optical axis (i.e. with $S(p) = 0$). In general, this approach works better for reconstructing the heights of small features with high precision, but may produce worse results for samples with occlusions.

S3.2 Reconstruction Dataset

To form the reconstruction dataset, the images are loaded, then cropped and normalized according to arguments specified by the user. The calibration maps are similarly computed and cropped.

First, the crop center, height, and width specified in normalized units are converted to pixels, then rounded to ensure compatibility with the U-Net (i.e., the crop height and width in pixels must be divisible by 32 when we are using a 4 layer U-Net with stride length of 2 in x and y).

Next, we need to identify the crop center for all the non-reference cameras. This is done to maximize the amount of field-of-view overlap between all the cropped images. For a given estimated height h_{est} (provided by the user, as described above) and center pixel for the crop from the reference camera p_{ref}, the center pixel for the crop from camera i would be:

$$p_i = p_{ref} + h_{est}[S_{ref}(p_{ref}) - S'_i(p_{ref})] - O'_i(p_{ref}) \tag{S6}$$

which is drawn from the forward projection equation, Equation 24. All the images are then cropped according to their individual crop center and the provided crop height and width. If the crop extends outside the bounds of a given image, the space is padded with zeros, and a binary mask is saved alongside the image to prevent the padded region from being used when computing loss.

Finally, the “inter-camera shift” and “shift slope” maps (S2) are generated, cropped, and re-normalized. The warp function used in this work requires maps to be normalized from (-1, 1) according to the size of the image being warped, so we begin this step by cropping the maps according to the center pixels p_i from above, then convert the shift amounts from units of pixels to normalized units according to the size of the cropped images. We then adjust the inter-camera shift maps to account for the differences between the crop centers for individual images. i.e. if $M_{c,i}$ is the cropped and normalized inter-camera map for camera i, we would adjust it as follows:

$$M_{c,i} = M_{c,i} - 2 \cdot (p_i - p_{ref})/L \tag{S7}$$

where L is the length (height or width) of the cropped image in pixels, and p_i and p_{ref} are the center pixel of the crop for image i and the reference image, respectively.

S3.3 Patching for Full Resolution Reconstruction

Height maps shown in this work were generated by running our reconstruction algorithm on 12, 24, and 48 GB GPUs. Even the 48 GB GPU is not large enough to reconstruct a full resolution height map over the full FOV and depth of an object in a single pass. The amount of space needed for reconstruction depends on a few factors:

1. The depth of the object. To achieve the highest quality results, we need small spacing between the planes in our back-projected volume (typically 50-200 μm when using full resolution images). This can result in prohibitively large volumes when working with objects with large height variation. For instance, to reconstruct over a 1 cm tall object with 100 μm spacing between planes, we would need $1\ cm/100\ \mu m = 100$ planes.

2. The number of pixels in the image.Cropping or binning the image reduces the number of pixels and the space needed on the GPU.

3. Architecture of the U-Net. While this was held fairly constant in this work, for some objects it may be possible to adjust the U-Net architecture to save space, without impacting the quality of the reconstructions.

4. The number of cameras used in the reconstruction. The number of cameras used has only has a small impact on the GPU space needed, since all images are summed into a single volume at the beginning of reconstruction. The individual images are only used when computing the loss, so the space required to store them is nearly negligible compared to the space required by the volume.

To achieve full resolution reconstructions over the full field-of-view of the system, we can run the algorithm on patches of the image, then tile these patches together to achieve a complete height map. The patching can be completed in one of two ways. In the first method, high resolution patches are reconstructed over the full field-of-view using the described algorithm, then tiled together. The number of patches needed when using this approach is dependent on the height variations within the sample. If the sample is relatively flat or has gradual height changes, we can use a smaller number of patches, as the height over which the back-projection volume must be formed can be reduced. However, if the object has sharp height changes, we will need to use a larger number of patches in order to allow for back-projection volumes with many planes. In future work, the process for selecting optimal patches for reconstruction can be automated, but as of this writing, it can be tedious to select the 3D bounds for each patch and reconstruction is time consuming.

In the second approach, we first compute a low-resolution height map over the full FOV. Then, high resolution patches are reconstructed using the low resolution map as a guide for the back-projection volume (see Equation 33). By using this approach, the number of patches needed in reconstruction can be drastically reduced, and the task of manually selecting patch bounds is eliminated. This is because we only need to search a small volume a few hundred microns above and below the estimated height in the low resolution map. With this approach, we were able to resconstruct the full FOV of an image with only nine patches, using a 24 GB GPU.

However, this approach has limitations. First, if the height value at a pixel was very inaccurate in the low resolution map, this will not be corrected in the high resolution version, since we only consider heights immediately above and below the originally estimated height. Second, very fine features that did not appear in the downsampled image may be missed in the higher resolution reconstruction, if they have significantly different heights from the surrounding features. Regardless, this approach is currently an efficient way to generate high resolution height maps over the full FOV of the system.

An example high resolution reconstruction is shown in Figure S7. We can compare the results using single-pass low resolution reconstruction with 4 x 4 downsampling, against a high resolution reconstruction made with nine patches. The high resolution reconstruction better highlights fine features, including small cracks in the knuckle and hairs on the skin surface.

S3.4 Noise reduction through summing

One benefit of our reconstruction approach is the ability to sum rectified images, $\hat{I}_i$, from all 48 cameras, to increase signal-to-noise ratio and contrast in the reference view. Additionally, since not all cameras are focused on precisely the same plane, the summed image may bring some features into focus which were blurred in the single image from the reference camera. Several examples of

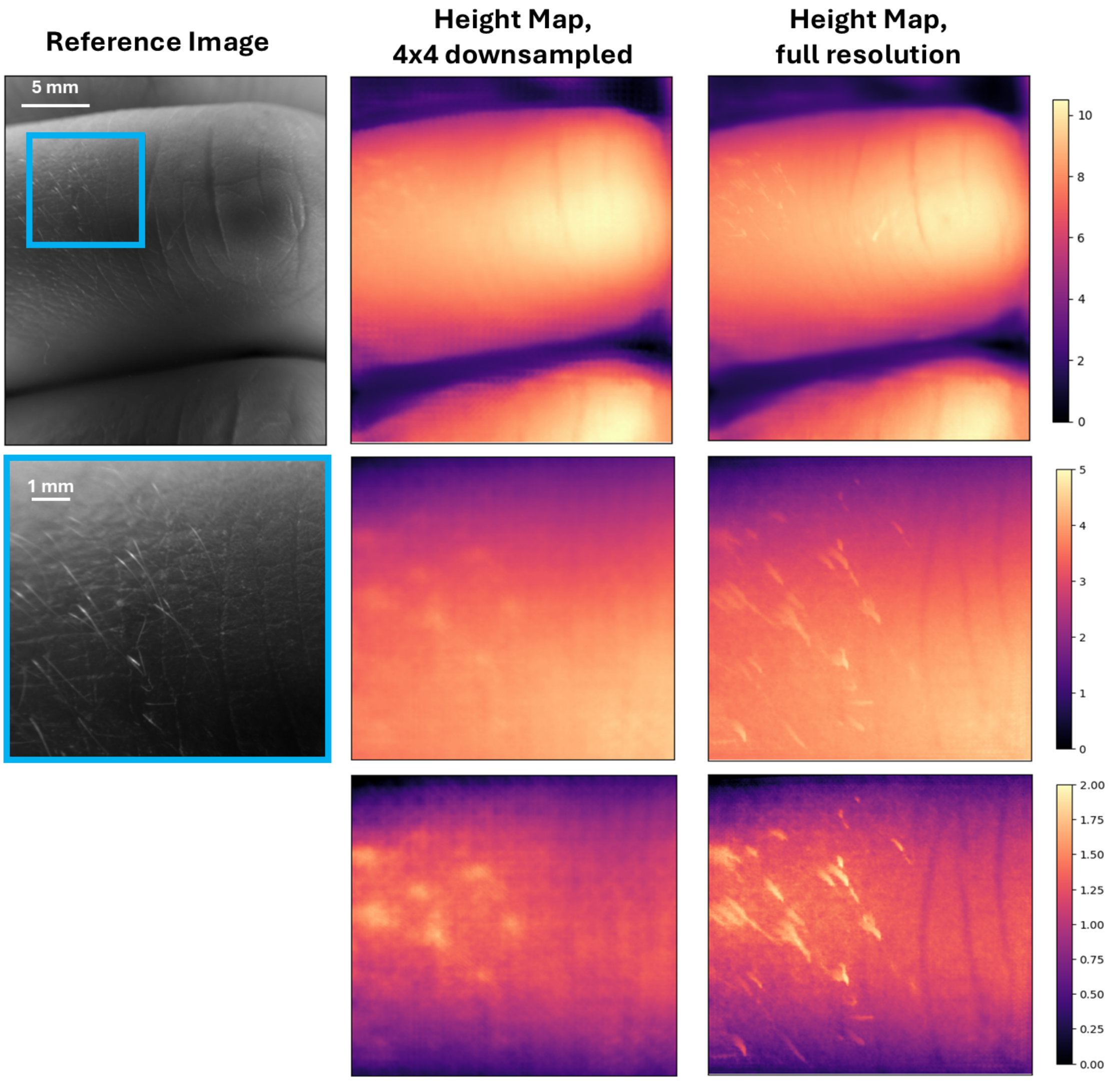

Fig S7 Patching The left column shows a grayscale image from the reference camera of a human finger. The center column shows the low resolution height map for the finger sample computed in a single pass with 4 x 4 downsampling. The right column shows the height map for the sample computed at full resolution, using patching. The top row is the full FOV image, the middle row is a cropped inset, and the bottom row shows the same inset after correcting for global tilt. We can see that the full resolution reconstruction includes finer details – emphasizing ridges and small hairs on the skin's surface. However, the large structure of the finger curvature remains the same between the low and high resolution versions. Scale bars are in millimeters.

this are shown in Figure S8.

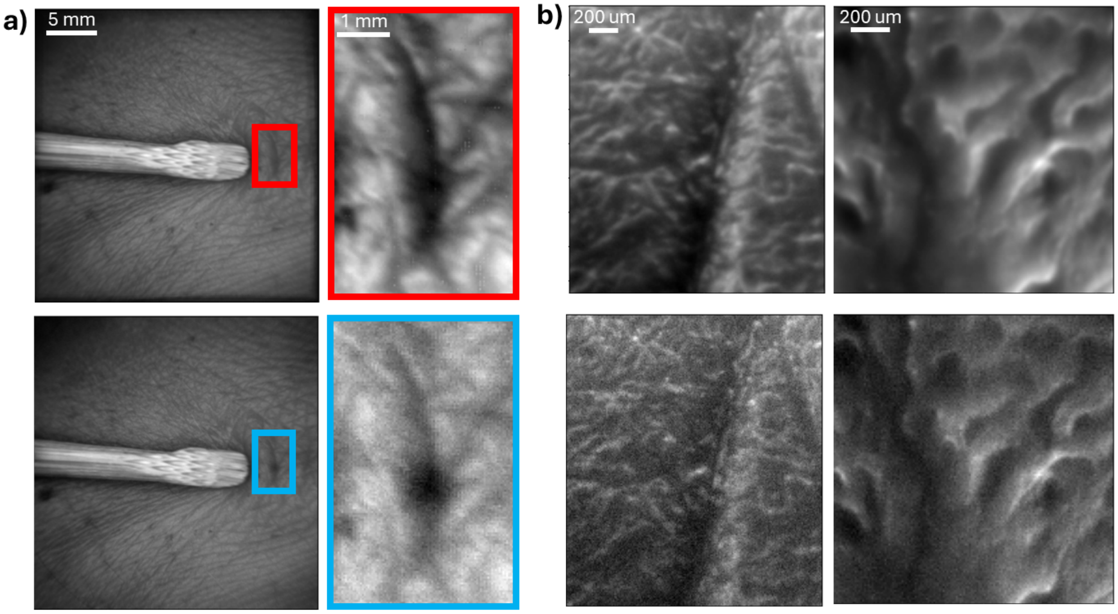

Fig S8 Noise reduction through summing. During the course of the reconstruction algorithm, we rectify all 48 acquired images to the reference viewpoint. By summing all 48 of the rectified images, we can form an image with improved signal-to-noise ratio and higher contrast. In both a) and b), the top row shows the summed rectified images, while the bottom row shows the single reference image **a)** Image of a human knuckle, acquired with 4 x 4 downsampling. In the bottom image, the portion of the skin highlighted in the inset was outside the DOF for the reference camera, creating a blurred image. By summing the 48 images, we can include information from cameras where this region was better focused and get a resulting image with better focus for the fine features in this region. **b)** Left: cropped patch of skin from the back of a human finger. Right: cropped portion of a rat skull. In both examples, the summed image has slightly decreased noise, allowing some fine features to be more easily visualized. At the same time, the focus is not as sharp in the top image.

S4 Multi-exposure Fusion

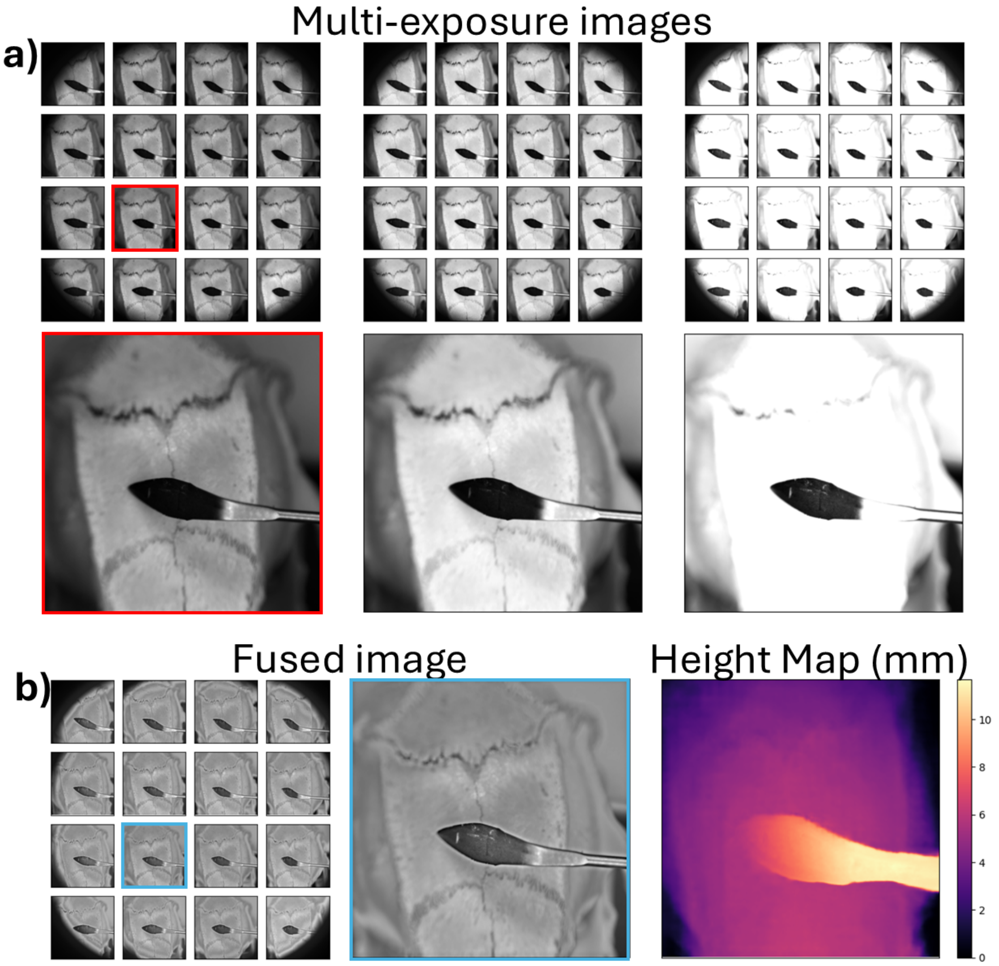

Fig S9 a) Images of ex-vivo rat skull acquired with a 16 camera FiLM-Scope, using three different exposure levels. b) Left: fused image using all three exposure levels together. Right: Reconstructed height map.

Here, we show how multi-exposure image fusion can be used to reconstruct scenes with large differences in reflectance and shadowing, as often occurs in surgical settings with reflective tissue and metallic tools. This was demonstrated using a modified FiLM-Scope setup with only 16 cameras and some vignetting in the edge cameras.

We first acquired three images of the same scene with different exposure levels (Figure S9a) and used the algorithm detailed in Mertens (2007)[59] to fuse the images separately for each of the 16 views, before using the fused images to generate a height map (Figure S9b). This approach produces reasonable results, although the reconstruction fails around sharp edges. For instance, the height map indicates that the handle of the tool is far wider than it appears in the image, because the fusion algorithm produces some ringing around these sharp edges. Moving forward, adapting the fusion algorithm or performing fusion on the images together during reconstruction could lead to improved results.

S5 Comparison with Optical Coherence Tomography

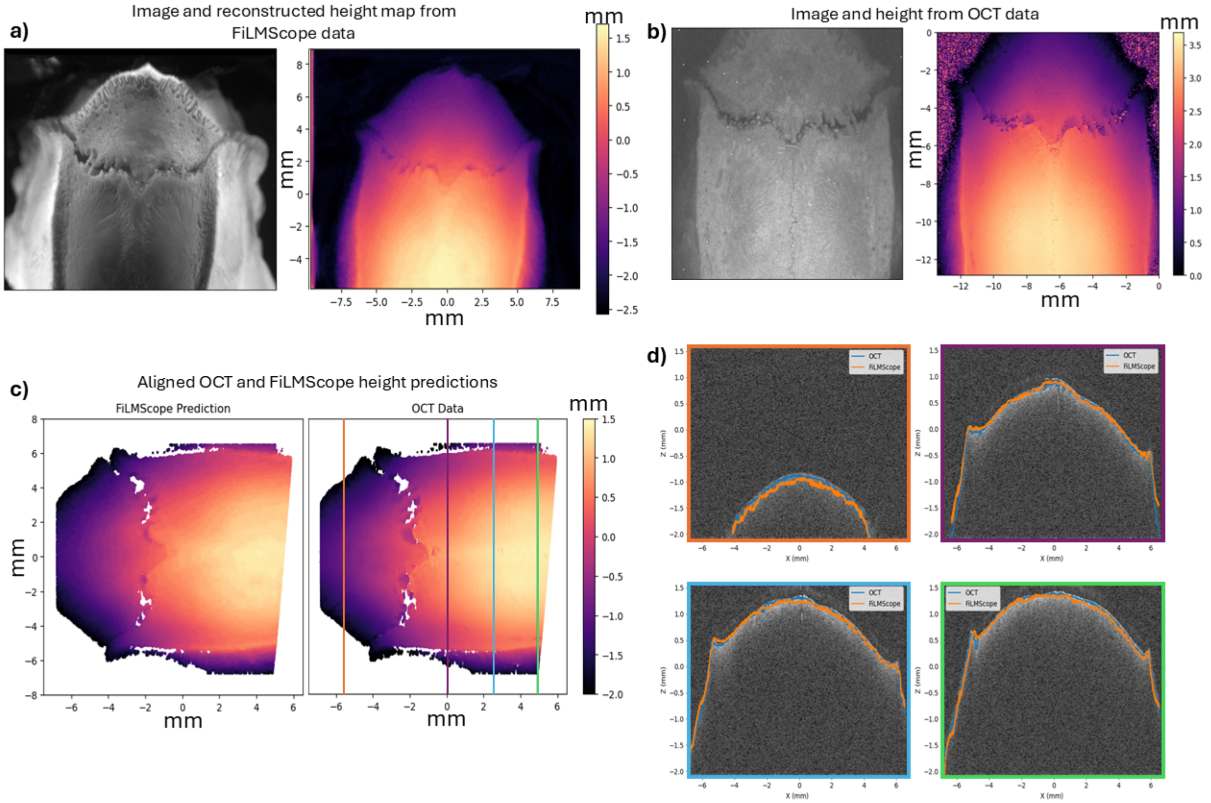

Fig S10 a) Image and reconstructed height map from FiLM-Scope system. b) Image and height map extracted from OCT volume. The image included the maximum intensity along each A-scan in the volume, and the height map indicates the indexed location of that maximum intensity. c) Aligned FiLM-Scope and OCT height maps. d) Cross sections from the OCT volume, at locations shown in c. Over each cross section we show the FiLM-Scope and OCT height maps at that plane.

To validate the accuracy of our approach, we imaged the same ex-vivo rat skull using both the FiLM-Scope and an Optical Coherence Tomography (OCT) system and compared the results.

The OCT engine consists of a Axsun Technologies 1060 nm swept-source laser (A12080125), with a scan depth of 3.7 mm. We scanned the skull using 1001 scan lines with 256 points each. After acquisition, we extracted a height map from the OCT volume by taking the index of the maximum point along each scan line. We then applied a binary mask to the height map to include only regions with adequately low variance, to ensure only the skull itself was considered, and not background regions. We acquired FiLM-Scope images of the same skull and performed 3D reconstruction over a region roughly corresponding to the region scanned by the OCT system.

We then aligned the height map from the OCT acquisition to the height map reconstruted from the FiLM-Scope images. This was done by coursely aligning using manually chosen key points, then refining the alignment using an iterative closest point algorithm implemented in the open source Python package *trimesh*. We then removed points from both the OCT and FiLM-Scope height maps that were not within 50 μm laterally of a point in the other map. The results after this alignment are shown in Figure S10c and d. **The root mean squared error between these two estimates was $33\mu m$.**

S6 3D Results from Known Objects

To test the accuracy of our reconstruction approach against objects with known heights, we imaged two 3D printed objects. The first was a cylinder printed on a MakerBot system (Figure S11a). The second is a pyramid printed on a Formlabs resin printer, then spray painted with white paint to create optical features (Figure S12a).

The results from the cylinder are shown in Figure S11. To test the reconstruction algorithm, we selected a patch from within the FiLM-Scope image, generated a height map, then estimated the radius of the cylinder from the height map. The true radius of the 3D model was 34.0 mm, and our predicted result was 34.1 mm, which is within the margin of error of the printer. After fitting the cylindrical shape, we subtracted the height of a perfect cylinder from our estimated height map, to visualize small height changes on the surface of the object (Figure S11d). By averaging this plot in the x and y directions, we can visualize the artifacts from the print filaments.

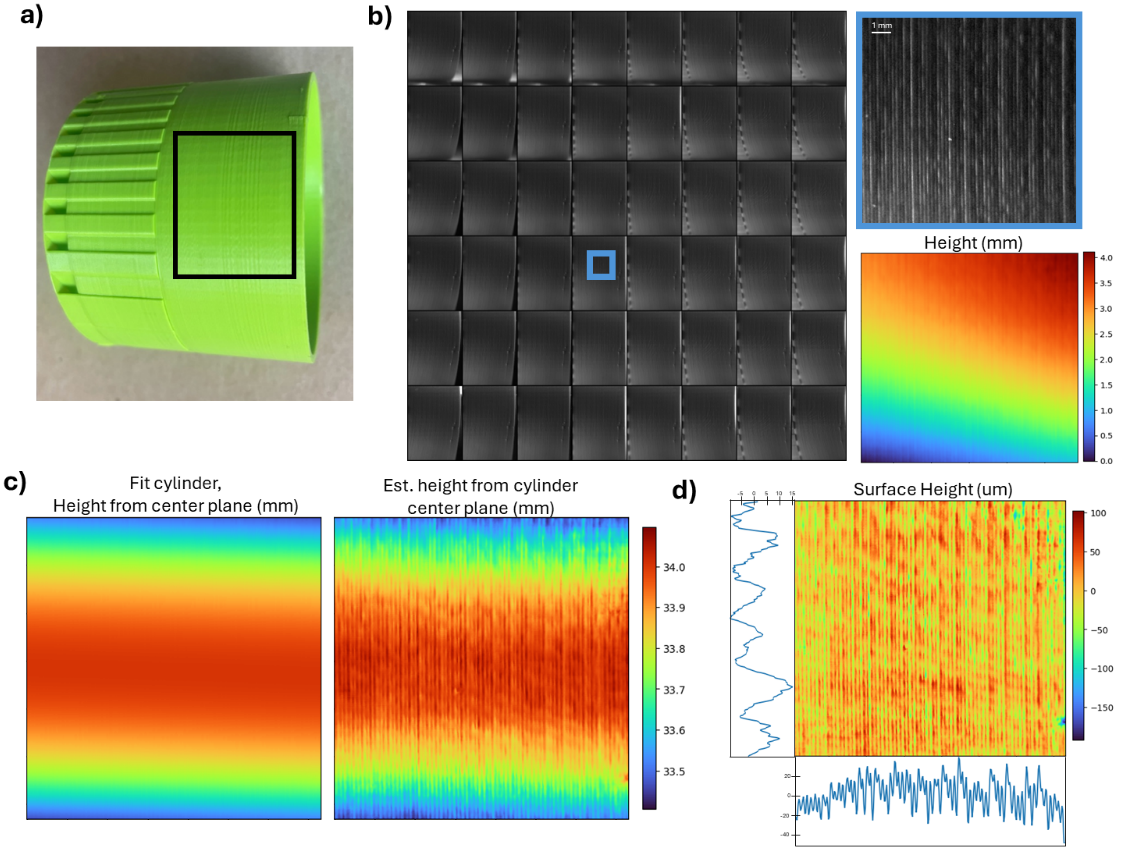

Fig S11 Ground truth sample: 3D printed cylinder. **a)** Picture of sample being imaged. The black box indicates the approximate region that was included in the FiLM-Scope image. **b)** Left: FiLM-Scope snapshot. Top right: inset from reference camera's image. Bottom right: Computed height map for the inset region. This height map was used to estimate the radius of the cylinder. Our estimate was 34.1 mm, while the true value was 34.0 mm, putting our error within the margin of error of the printer itself. **c)** We converted each pixel's height to be the distance of that point from a plane intersecting the cylinder, accounting for the tilt of the cylinder relative to the FiLM-Scope. The right height map shows the estimated height of the imaged cylinder, while the left map shows the generated height for a perfect cylinder with a radius of 34.1 mm. **d)**. By subtracting the left height map in c from the right height map, we can see the height of the surface of the cylinder. The plots to the bottom and right of the height map are the results of averaging the map across its rows and columns. In these plots, we can see the printer artifacts. Particularly, in the bottom graph, we can see the heights of individual filaments.

The results from the pyramid are shown in Figure S12. For this sample, we reconstructed over the full FOV with 4 x 4 downsampling, then performed 2D gaussian filtering on the height map, with σ = 4 pixels. The portion of the pyramid in the image FOV is over a centimeter tall, well outside our reported DOF of 3 mm, but we are still able to achieve a relatively accurate height reconstruction over its full depth. The height profile within the white box in Figure S12b, is plotted in the top panel of Figure S12c. We can clearly see the triangular shape over the full depth extent of the object. We then fit a line to the portion of the triangle highlighted in red. The bottom panel of Figure S12c shows the portion of the pyramid to the right of the red line, after subtracting out the fit line, with estimated height on the x-axis and the difference between the estimated height and the line of best fit on the y-axis. From this plot, we can see that as the object falls outside our reported DOF, the accuracy of the estimated height reduces significantly. It is worth noting that the trend is not random – the estimated values deviate further and further below the true value as the object is further outside the DOF. Thus, in the future, we can measure this trend and compensate for it in our calibration method, allowing us to achieve more accurate 3D results over a larger depth range.

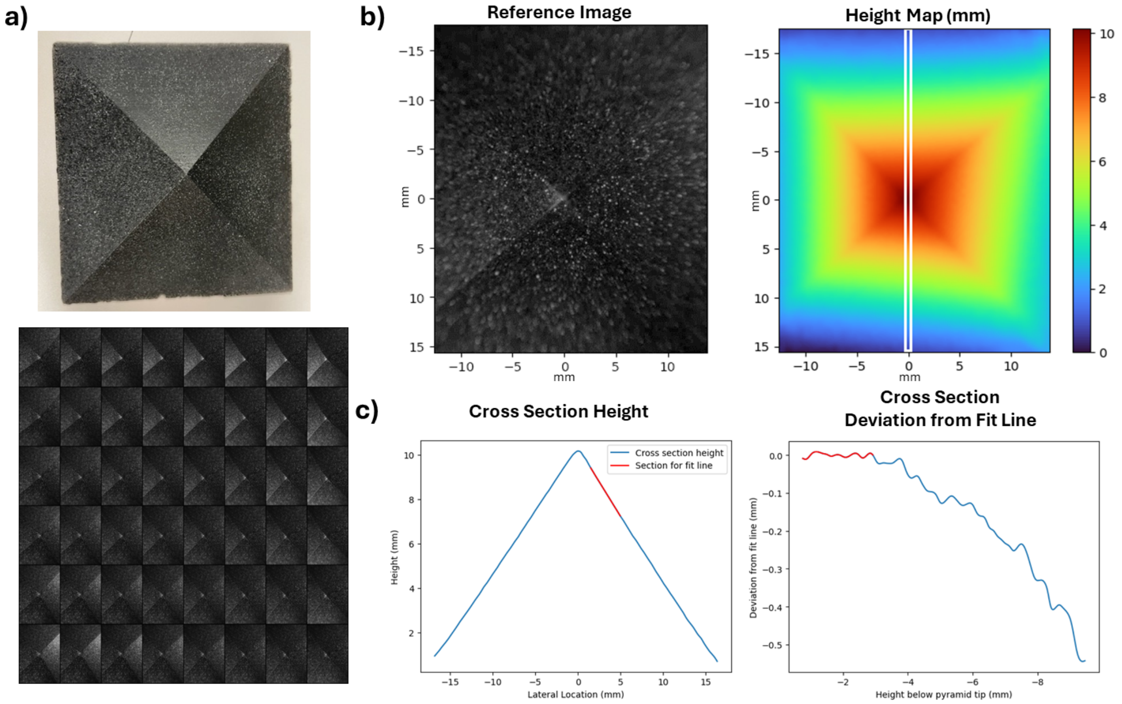

Fig S12 Ground truth sample: pyramid. a) Top: sample to be imaged. This is a pyramid printed on a FormLabs resin printer, then finely painted with white spray paint to add optical texture. Bottom: FiLM-Scope snapshot of pyramid. **b)** Left: Single image of pyramid. Note that the pyramid is extending outside the DOF of the FiLM-Scope around the edges of the FOV. Right: Estimated height map of the pyramid. We achieve a reasonable estimate of the pyramid shape, even outside the bounds of our reported DOF. **c)** Cross sections of pyramid height map. Left: Plot of cross section marked in white in b). The cross section is averaged over a width of 11 pixels. We used the portion of the plot highlighted in red to fit a line to the pyramid's slope, to compare how far our height estimation deviated from the expected linear slope. Right: Difference between the line of best fit and the estimated height of the cross section. Note the x-axis is now "height below the pyramid tip". From this plot, we can see that as the object extends past the DOF of the system, the estimated height becomes biased in one direction. In future versions of the reconstruction algorithm, we can adjust the calibration model to account for this bias.

S7 Sensor Synchronization

Ensuring high levels of synchronization between cameras is important in avoiding motion artifacts when reconstructing 3D video of moving objects. The FiLM-Scope uses a version of the multi-camera array microscope (MCAM),[44,45] which synchronizes the micro-cameras using a shared FPGA. All of the sensors (AR1335, OnSemi) in the MCAM sit on a single PCB board and are triggered by the same signal from the FPGA. As a result, any delays in the start of image acquisition are due to latency in the PCB itself, which is estimated to be well under 1 μs.

The sensors use rolling shutter image acquisition, which we can use to demonstrate the level of synchronization. We used an output trigger signal from the FPGA to flash an LED after the start of image acquisition. For this test, we used color image sensors with a GBRG bayer pattern, so we only considered every other row in the sensor, to ensure consistent light collection. When acquiring full resolution (3072 x 3072 pixel) images, the rolling shutter takes 96.8 ms, so the time between the start of each considered row is $96.8ms/(3072/2) = 63\mu s$.

An example result from these tests is shown in Figure S13. While the precision of this test is limited by the row readout time of the sensor, we can see that the level of precision is well under the 63 μs we can confidently claim using this approach.

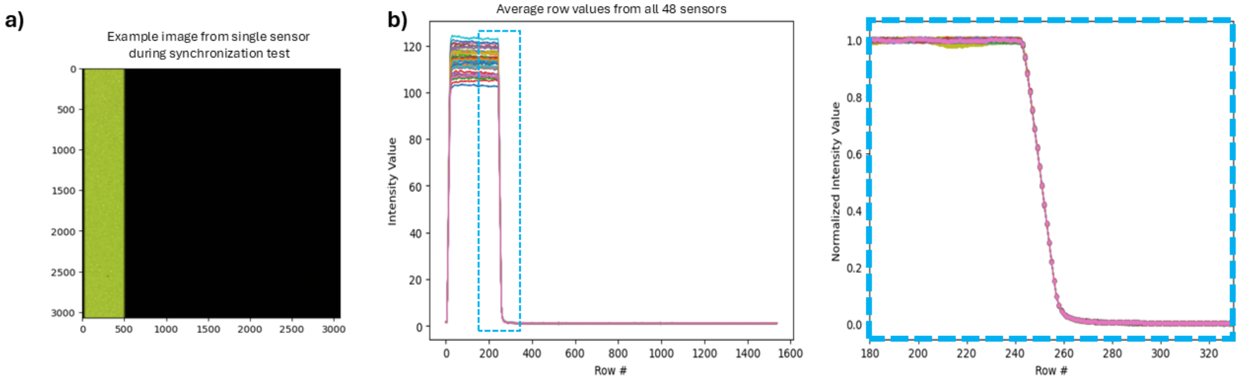

Fig S13 Results from synchronization tests with FiLM-Scope sensors. a) Example image from a single sensor during this test. Due to the rolling shutter of the sensor, only a portion of the sensor recorded signal from an LED triggered immediately after the start of image acquisition. We can determine the time at which the sensor began image acquisition from which rows detected signal. b) Row intensity values from all 48 sensors during a single test. These values were computed by averaging values along every other row of the sensor, giving a total of 3072 / 2 = 1536 rows. Left: raw intensity values across all rows. Right: Cropped portion of the sensor, with intensity values normalized for their maximum value on each sensor. From this image, we can see that the sensors are very tightly synchronized, well under the 63 μs readout time between rows.

S8 System Illumination

Because 3D information encoding in Fourier light field imaging does not depend on a specific illumination scheme, the FiLM-Scope can be used with a wide range of illumination setups. In the future, this could include darkfield or fluorescent imaging. In this work, we used two different epi-illumination setups. The first was exterior epi-illumination with a flexible LED strip fixed to the front of the lens (Figure S14a). The second was internal epi-illumination from a large ring illuminator placed directly behind the primary lens (Figure S14b). Because of the limited working distance with our primary lens, it was difficult to evenly illuminate over the full surface of an object using the LED strip lighting. The ring illuminator provided far more even illumination, but left significant illumination artifacts on the images, as shown in Figure S14c. To compensate for this, before imaging, we acquire an illumination correction image with the illumination source turned on, but without a sample. We can then perform background subtraction to reduce the illumination artifacts in our final images. This approach does limit the dynamic range of the impacted pixels, and if the illuminator shifts during imaging, we cannot directly subtract the correction image from the acquired images. In the future, we will work towards an improved illumination setup, possibly by using crossed polarizers to reduce reflection artifacts from the illumination source.

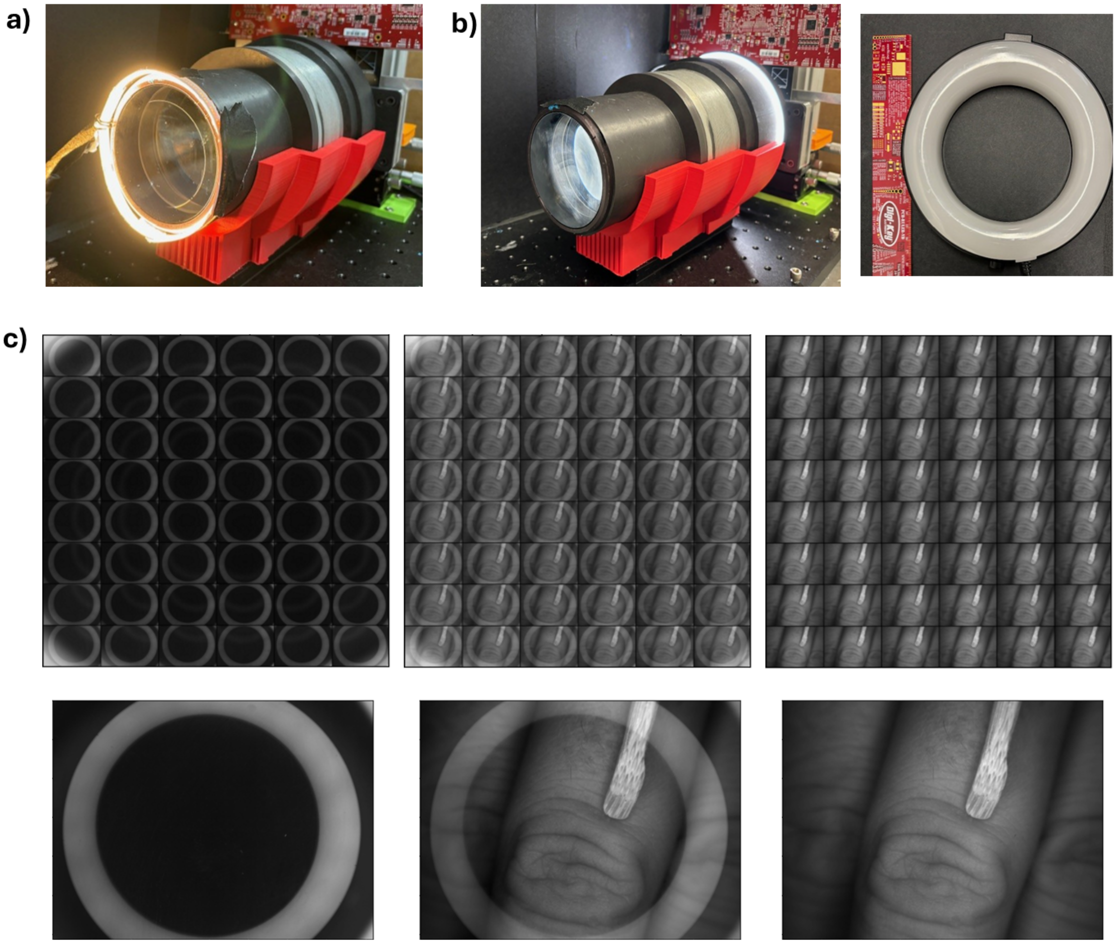

Fig S14 Illumination. Two different illumination setups were used for the images shown in this work. **a)** Illumination setup 1: external epi-illumination via flexible LED strips fixed to the front surface of the lens. **2)** Illumination setup 2: internal epi-illumination via a ring illuminator placed between the primary lens and the lens array. **c)** While internal epi-illumination provides more even illumination over the sample surface, it produces significant artifacts. The left panel shows these artifacts with no sample present, and the middle panel shows a snapshot of a human finger including these artifacts. Before reconstruction, the artifacts are removed from the snapshots, by subtracting the left set of images from the middle set of images. The result is shown in the right panel. However, this still limits the dynamic range of the affected pixels, so future versions of this system should have an improved illumination setup.

S9 Camera Ablation Study

In order to inform the design of future FiLM-Scopes optimized for specific surgical or non-surgical applications, we studied the impact of the placement and number of cameras on 3D reconstruction for a given sample. In general, the number of perspectives needed for successful 3D reconstruction of an object from multi-view images is both algorithm and sample dependent. Very dense or complex objects, particulary non-convex objects, may require many such images, while simple or sparse samples (such as triangulation of a small number of fluorescent particles), may be done with as few as 2 or 3 perspectives.[41] In addition to the absolute number, the placement of cameras is critical for successful 3D reconstruction. The axial resolution of the system is proportional to the angular separation of cameras (see Table 2 in the main text), so a small number of angularly separated cameras can allow for high resolution 3D reconstructions for simple samples. In the FiLM-Scope architecture, there are drawbacks to imaging with a large number of cameras - streaming from many cameras leads to lower frame rates and requires a primary lens with a larger diameter, increasing the size and weight of the system. Thus, it is important to carefully consider the number and placement of cameras when designing a system for a particular use case.

To demonstrate how the number and arrangement of cameras used can impact the accuracy and precision of 3D reconstruction, we used the FiLM-Scope to acquire images of two samples: a small rock with a textured surface and significant height variation, and a thin microstamp covered with 50 μm tall pillars. For each sample, we performed multiple 3D reconstructions from the acquired snapshot, using a different subset of cameras for each run, and compared the results.

For the rock sample, we chose five patches of 100 x 100 pixels from within the image, and performed 3D reconstruction on each of these patches with eight different subsets of the 48 cameras, including with all 48 cameras to serve as a pseudo-ground truth. The errors between these height maps and the pseudo-ground truth height maps are summarized in Figure S16b. As expected, when we used smaller subsets of cameras with worse theoretical axial resolution, the MSE of our reconstructed height maps increased. For subsets with a small number of cameras but large angular disparity between views (and improved theoretical axial resolution), some patches achieved 3D reconstructions with low MSE compared to the pseudo-ground truth, while other runs partially or fully failed to converge, leading to very inaccurate height maps (Figure S16b). It is worth noting that those results could likely be improved by using an alternative method to provide an initial estimate of the object's height, such as using depth-from-focus cues or performing a preliminary height reconstruction at a lower resolution. These results indicate that having many cameras may be useful for objects with large height changes and complex surfaces.

For the microstamp with 50 μm pillars, we performed 3D reconstruction on a single 608 x 608 pixel patch from within the image, using eleven different camera subsets. From each reconstruction, we pulled height values from 24 lines of roughly 24 pillars each, and identified the peaks and troughs along these lines to find the estimated heights of the pillars. We then created a histogram of the peak and trough height distrubtions, and assessed the standard deviations of the two distributions (which would ideally be small), and the distance in micrometers between the center of the two distributions (which should match the height of the pillars, 50 μm). This procedure is shown in Figure S15.

We repeated this procedure for several camera subsets, 9 of which are shown in Figure S17. First, in the top row, we compared camera subsets with the same maximum stereo sepearation, but different numbers of cameras. Unlike the rock sample, reconstructions with angularly separated sets of down to four cameras showed comparable results to the reconstruction with all 48 cameras.

Quality began to decline when we used only three cameras. This is likely because the sample is nearly flat, so the algorithm can converge more easily while still differentiating fine features on the surface. In the bottom row, we considered gradually more compact camera subsets of adjacent cameras, shown in the top row of Figure S17. Similar to the rock sample, we see gradually worse performance for smaller subsets of adjacent cameras: the reconstruction with all cameras shows an average separation between peak and trough heights of 46 μm, while the reconstruction with 3 x 3 cameras yields a less accurate estimate of 79 μm. In the reconstruction from 2 x 2 cameras, we were not able to successfully identify peaks and troughs in the extracted lines.

Going into the future, we can test the accuracy of 3D reconstructions from the FiLM-Scope on specific tissue types, to determine the architecture that is needed for different surgical settings.

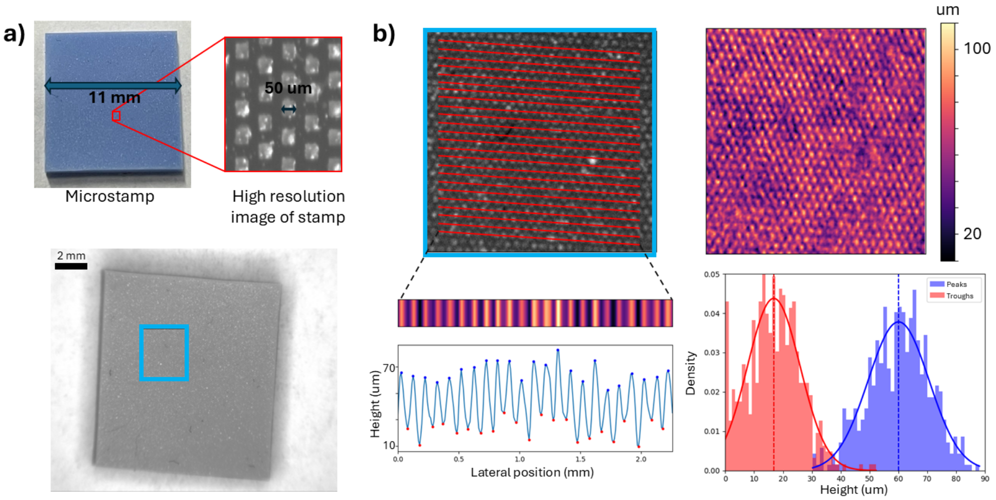

Fig S15 Performance assessment with microstamp. **a)** Microstamp sample. The surface of the stamp is covered with 50 μm tall pillars, spaced 100 μm apart. Top: image of microstamp showing width of pillars. Bottom: FiLM-Scope image of the microstamp. 3D reconstructions were performed with different camera subsets on the region shown in the cropped inset. **b)** We reconstructed a height map for the microstamp (top right) then extracted the heights of 24 rows of pillars, highlighted in red in the image on the top right, and identified the peaks and troughs along those lines. The estimated height along a single row of pillars is shown in the bottom left plot. Peaks are marked in blue, and troughs are marked in red. We can then histogram the peak and trough heights to assess system performance (bottom right).

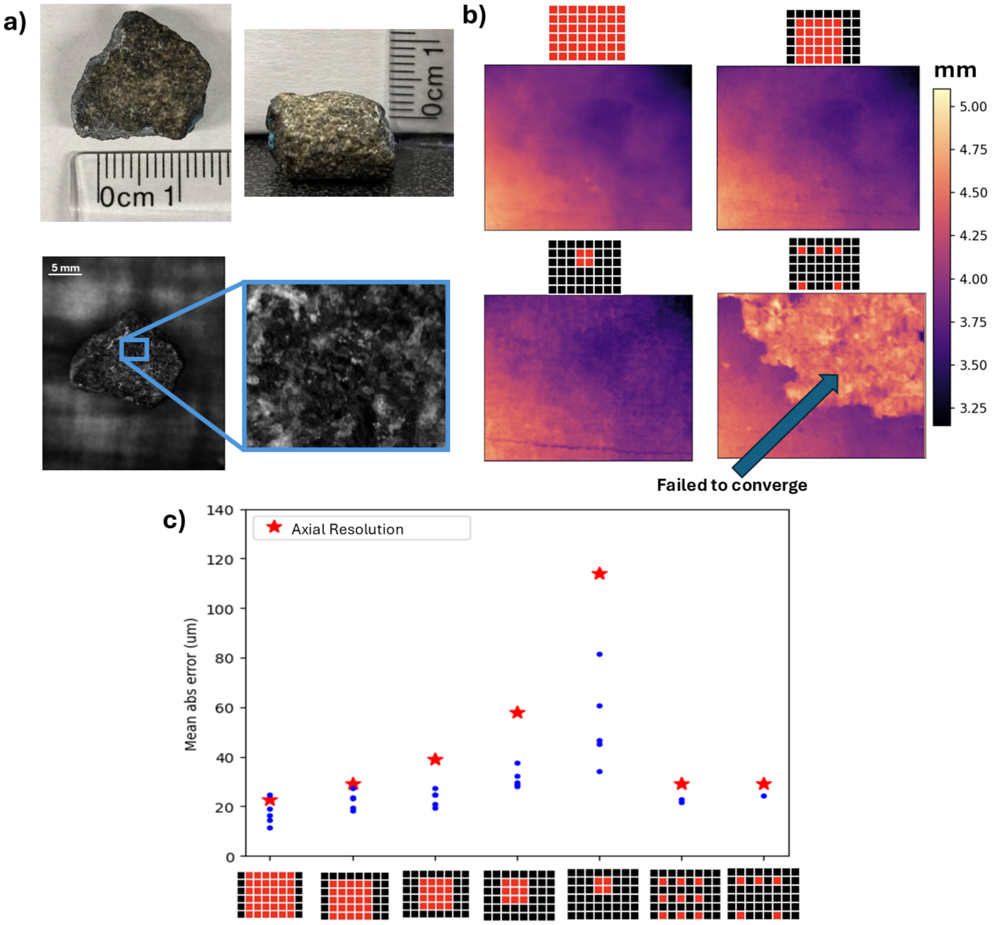

Fig S16 Camera ablation with rock sample. This figure shows the results of performing reconstructions with different numbers of cameras. The red and black grids indicate which cameras were used in a given reconstruction. Cameras shown in red were used, while those in black were omitted. **a)** Image of rock sample. Top: top and side view of rock. Bottom: FiLM-Scope image of rock. The inset shows an example patch of the image on which reconstruction was performed. We repeated the 3D reconstruction with each camera subset for five such patches.**b)** Example height maps of a single patch for four camera subsets. The pseudo-ground truth reconstruction done with all 48 cameras is on the top left. The reconstructions with 5 x 5 and 2 x 2 cameras (top right and bottom left, respectively) retain the same general height map, but both are noisier than the reconstruction with all cameras. On the bottom right, the reconstruction with five spread out cameras has a large region that failed to converge, resulting in a very inaccurate height map.**c)** Using the reconstructions with all cameras as a pseudo-ground truth, we took the mean absolute error between the pseudo-ground truth height map and the height map reconstructed from each camera subset. We can see that for the first five subsets, the errors increase with increasing axial resolution. For the two camera subsets on the right of the plot, some patches achieved reconstructions with very low error. However, others not included on this plot failed to converge.

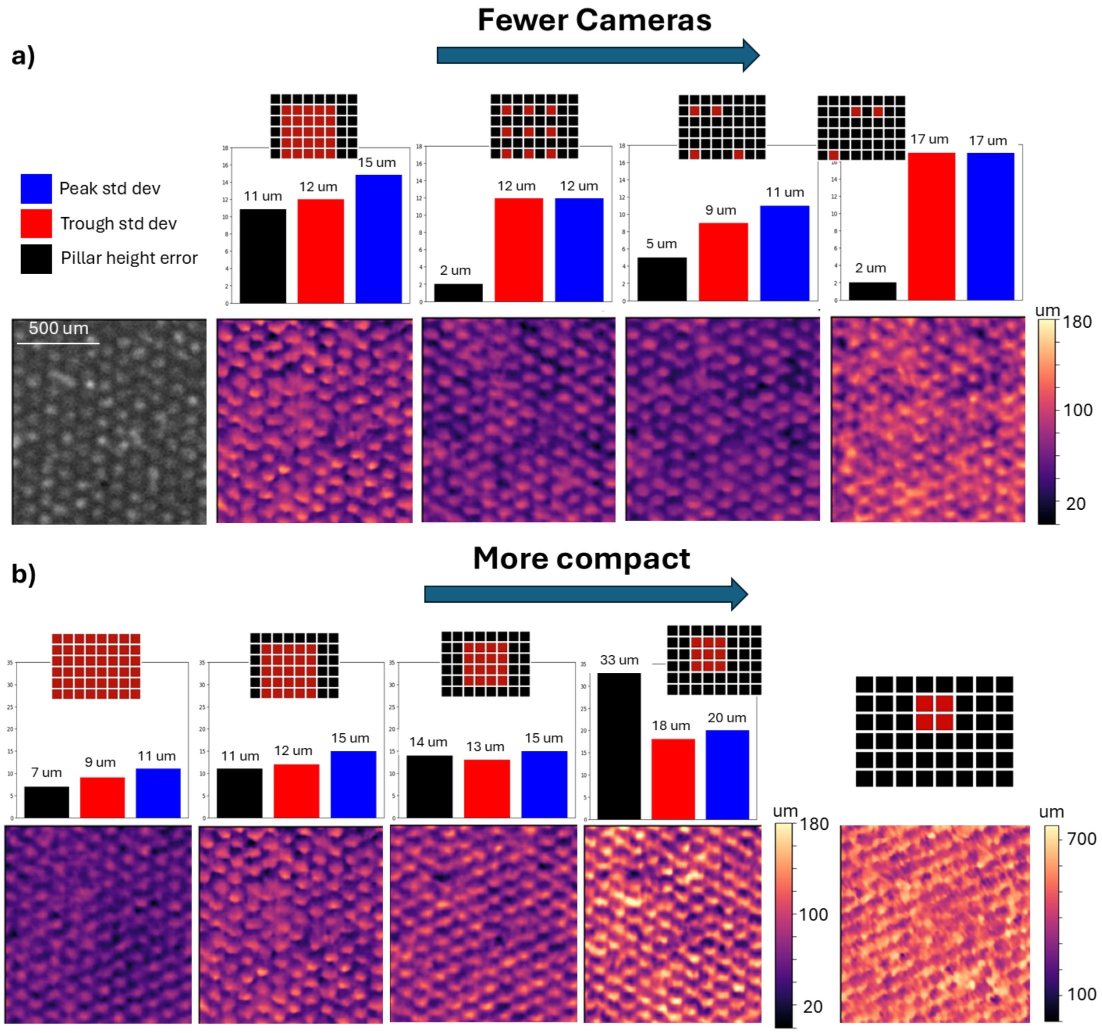

Fig S17 Camera ablation study with microstamp In this figure, we repeated the procedure shown in Figure S15 and show the results for 9 different camera subsets. a) Comparison of camera subsets with different numbers of cameras but equal stereo disparity. We see similar height map quality from 25 down to only 4 cameras. There is a noticeable drop in precision when only using 3 cameras. b) Comparison of gradually more compact subsets of adjacent cameras. 6x8, 5x5, and 4x4 camera subsets all retain high quality 3D reconstructions, with a noticeable drop in quality for 3x3 cameras. The reconstruction with 2x2 cameras failed to converge, and we cannot distinguish individual pillars in the resulting height map.