Supplementary information

S1 System characterization: lateral resolution, axial precision and accuracy, and depth of field

We performed several experiments to characterize the performance of our computational 3D imaging system, starting with imaging of a USAF resolution target near the center and edge of the field of view (FOV) of a single camera (Fig. S1a,b). Our system can resolve group 5 elements 2-3, corresponding to a bar width of 12-13 µm or a full-pitch lateral resolution of ∼25 µm. We then characterized the depth of field (DOF) by axially translating the same flat patterned target used in Figs. 4 and 5, using a motorized stage (Zaber) in increments of 0.25 mm. This defines the axial FOV of our 3D reconstructions. For each axial position, we computed a contrast metric based on the mean image gradient magnitude (Fig. S1c). The full width at half maximum (FWHM) of this curve is 9.434 mm, which is similar to value obtained by fitting the curve to the intensity of a Gaussian beam,

$$I(z) = \frac{I_0}{1 + \frac{(z-z_0)^2}{z_R^2}} + I_b, \tag{S1}$$

where I_0 and I_b are the arbitrary amplitude and offset, z_0 is the focal position, and $2z_R$ is the DOF, corresponding to when the lateral resolution degrades by $\sqrt{2}$. Least-squares fitting yields $2z_R = 9.402$ mm. In practice, the DOF may be smaller if the neighboring cameras are not focused to the same plane, such that the focus regions are offset.

Finally, we characterized the accuracy and precision of our 3D height maps by imaging 6 gauge blocks (Mitutoyo), precisely machined and characterized to be within 0.3 µm of their nominal values: 1.000, 1.020, 1.050, 1.100, 1.200, and 1.400 mm (Fig. S1d,e). We computed the accuracy as the absolute error between the estimated and ground truth heights, aggregated across all pixels within each gauge block, and the precision as the standard deviation of the height estimates across each gauge block, which are summarized in Table S1 for all three configurations in Table 1. Since there is an arbitrary global height offset, we chose the one that minimizes the MSE between the estimated and ground truth heights [1].

S2 Generalization experiments

Here, we show that the multiocular stereo CNN trained on a subset of frames can generalize well to unseen frames. As validation we compare this generalization performance to that of a monocular stereo CNN (i.e., one that only takes in a single image as the input). To make these comparisons, we picked two independent subsets of the video frames. In Set 1, we took about 15 frames

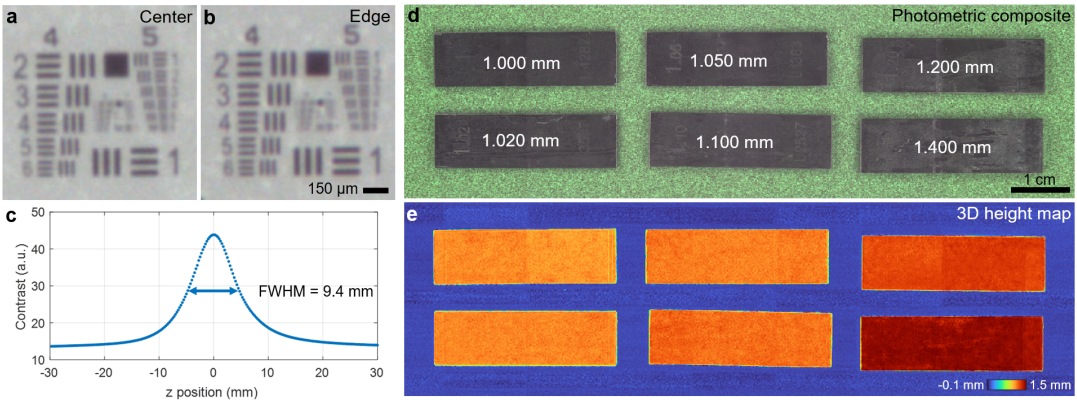

Fig. S1 System characterization experiments. **a**, **b**, USAF resolution test chart image near the center and edge of the FOV of one camera without downsampling. **c**, Image contrast of a patterned target as a function of axial position. **d**, Stitched photometric composite of 6 precisely-machined gauge blocks placed on a green patterned target (captured with the 60-fps configuration), with their nominal thicknesses denoted. **e**, The reconstructed 3D height map of the gauge blocks. Accuracy and precision are quantified in Table S1.

Ground truth height	1× downsamp		2× downsamp		4× downsamp	
	Acc.	Prec.	Acc.	Prec.	Acc.	Prec.
0	44.3	19.3	25.3	17.2	60.0	55.9
1000	8.9	17.5	12.0	32.2	50.6	69.4
1020	4.1	11.2	18.1	24.3	51.6	72.7
1050	3.2	18.5	4.3	25.7	14.8	63.8
1100	7.9	17.7	7.8	28.6	12.7	68.7
1200	5.2	24.1	0.4	33.0	20.3	88.9
1400	55.0	8.7	1.0	27.7	49.4	100.4
mean	18.4	16.7	9.8	26.9	37.1	74.3

Table S1 Accuracy (absolute error from ground truth) and precision (standard deviation) of the height estimation of the 6 gauge blocks (and background) in Fig. S1a,b for all three downsampling configurations. All values are in µm.

equally spaced temporarily across the video. In Set 2, we took another 15 equally spaced frames at half a period offset with respect to Set 1. For example, if the video was 601 frames, then Set 1 would consist of frames 1, 41, 81, ... 561, 601 and Set 2 would consist of frames 20, 60, 100, ...540, 580. We then trained two independent multiocular CNNs, one on Set 1, the other on Set 2, and compared the 3D height map predictions on both sets. The idea is that in the absence of ground truths, the physics-supervised CNN predictions on training set examples could serve as pseudo-truths. For comparison, we also trained a monocular CNN on Set 1 and compared predictions on Set 1 and Set 2.

Figs. S2 and S3 show the comparisons among these three CNNs for both zebrafish and fruit flies. In both organisms, the multiocular CNNs generalize well to unseen video frames, based on comparisons between images from the CNN trained on Set 1 and the one trained on Set 2. However, for zebrafish,

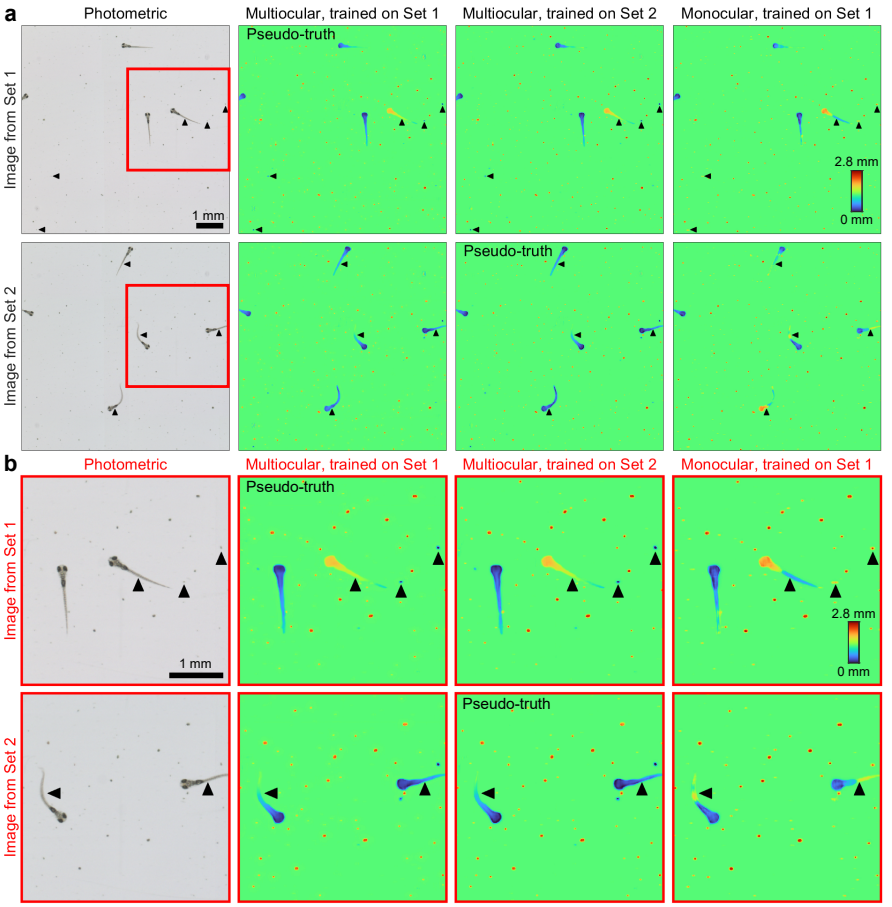

Fig. S2 Generalization performance of multiocular and monocular CNNs trained on frames from a video of freely swimming zebrafish. **a**, First row shows an example from Set 1 and 3D height predictions of three different CNNs – two multiocular CNNs, trained on Set 1 and Set 2, and one monocular CNN trained on Set 1. Second row shows predictions on Set 2. **b**, Zoom-in of the red boxes in **a**. Arrowheads point out features for which the multiocular CNN generalized well, but not the monocular CNN, as evaluated by comparing the predictions the respective pseudo-truth.

the monocular CNN (trained on Set 1) generalizes poorly (to Set 2). This is evidenced by erroneous heights of several zebrafish's heads or tails, as it is difficult to determine the heights of the fish based on appearance alone – magnification-based cues are confounded by natural size variation. Similarly, the monocular CNN incorrectly estimates the heights of the sunken food particles. This is likely due to the fact that the vast majority of food particles are floating, and since the food particles have no discernible height indicators, the monocular CNN simply uniformly assigns the floating height to all particles. While the monocular CNN performs better for the fruit flies than for zebrafish, it still makes a few errors, e.g., when one fly is climbing on top of another.

Such fly behavior was rare in our captured video, so the monocular CNN had fewer training examples to learn the semantic cues to accurately predict the elevated height, whereas the multiocular CNN was able to predict the elevated height from the parallax cues.

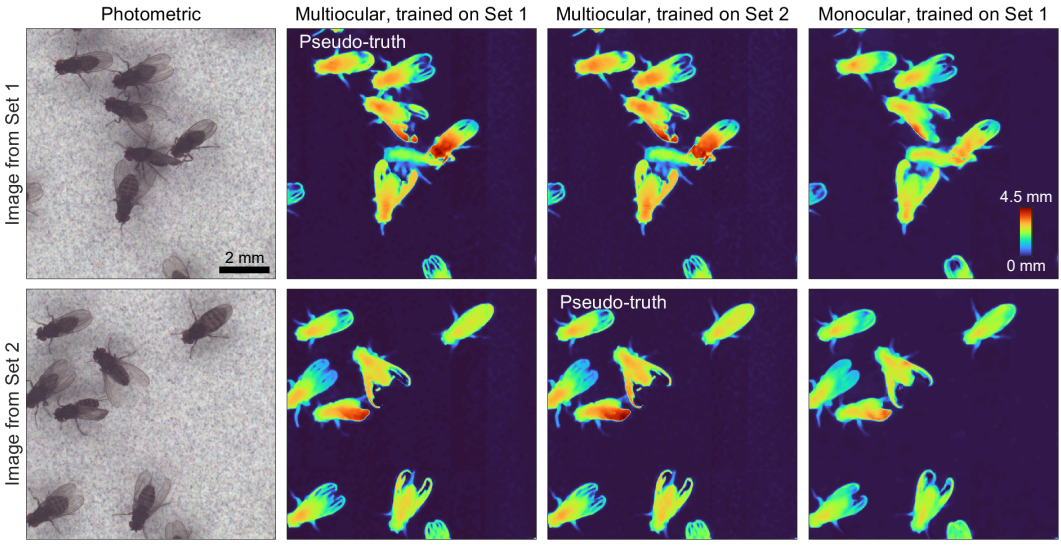

Fig. S3 Generalization performance of multiocular and monocular CNNs trained on frames from a video of fruit flies. First row shows an example from Set 1 and 3D height predictions of three different CNNs – two multiocular CNNs, trained on Set 1 and Set 2, and one monocular CNN trained on Set 1. Second row shows predictions on Set 2.

S3 Implementation details on patch-based training with multi-ocular stereo inputs

As mentioned in the main text (Sec. 2.4), the CNN is supplied multi-view inputs of the same sample scene (as shown in Fig. 2a,c), whose goal is to improve the generalizability of the CNN (Supplementary Sec. S2). These neighboring views are stacked along the channel input dimension in a way that preserves convolutionality, so that patch training and full-FOV inference are consistent. This is beneficial because monocular stereo depth estimation is insufficient for objects whose appearances don't change significantly as a function of depth. For example, when imaging a fruit fly or zebrafish larva, it is difficult to distinguish between height-dependent magnification changes and natural variation in organism size. Thus, we train our CNN to solve a multi-ocular stereo 3D estimation problem, which is better-posed, as the 3D supervision signal itself is derived from the registration of the multi-ocular data (Supplementary Sec. S2). In this paper, we use 3 stereo inputs or fewer (center, left, and right, if available).

Here, we expand upon the explanation of our patch-based CNN training procedure given in Sec. 2.5 and Fig. 2c.

S3.1 Determining the observing cameras and the coordinates

We start with the camera pose calibration based on a flat patterned target (Methods 5.3) to generate a "visitation log", V. V is an $n_{row} \times n_{column} \times 54 \times 2$ tensor look-up table specifying which of the 54 cameras view a certain spatial position in the reconstruction coordinate system as well as the respective (row,column) pixel coordinates in the camera coordinate system that map to that position. The formation process of V is somewhat similar to the backprojection step of the reconstruction (Fig. 2a), but instead of backprojecting the RGBH values, we backproject the (row, column) coordinates. This visitation log facilitates rapid retrieval of the relevant cameras for each randomly sampled position. Note that since we want to avoid rolling shutter artifacts that may occur where the bottom of one camera overlaps with the top of the camera below (Methods 5.1 and Supplementary Sec. S4), we only consider horizontal overlap.

S3.2 Selecting random patches

Given this visitation log, we select n_{batch} random 2D coordinates in the reconstruction frame of reference for each CNN training iteration. For each of these random coordinates, we retrieve the relevant cameras and their corresponding camera-centric coordinates. For each camera image, we then crop out a square patch of width w_{patch} centered at the sampled coordinates. If these coordinates are within $w_{patch}/2$ of a camera image edge, they are shifted so that the patch remains within the image.

For each image patch, we also extract patches from the left and right cameras and stack them along the channel dimension of the CNN input, which the CNN can exploit for 3D estimation (Fig. 2c). To do this in a manner consistent with both training on patches and inference on full-sized images, we homographically transformed the left/right neighboring images into the frame of reference of the central camera in question, as if the sample were flat (more precisely, coincident with the pre-calibration reference plane; Sec. 2.3, Methods 5.3). If the sample were completely flat, then the transformed neighboring images would theoretically be identical to the image captured by the camera in question where their viewpoints overlap. However, if the sample exhibits height variation, the transformed neighboring images would exhibit parallax shifts in proportion to the height variation. When there is no left or right camera (i.e, the first or last column of cameras), we input blank images (all zeros). Similarly, when either the left or right patch overlaps with the edge of its respective camera, we assign zeros to the missing regions. Note that in this scenario, we cannot shift the left/right patch away from the edge, as we could above, because the left/right patch must remain coaligned with the main

(central) patch so that we maintain full convolutionality for the inference step (Supplementary Sec. S3.8). Furthermore, we do not want to exclude training cases where the central patch is close to the edge of the camera, as these cases appear when applied to full-size camera images during the inference step.

We note that the number of cameras observing a particular point can range from 1 - 3, since we only consider horizontal overlap. When only one camera views a particular point (the left and right edges of the reconstruction) during training, we reject the resulting patch as there's nothing to register. To account for the fact that the number of patches may vary for each batch element, we use tensorflow's [2] `tf.RaggedTensor` construct, which allows some dimensions of a tensor to have slices with different lengths. In our experiments, we used n_{batch} =1, 2, and 8 for the 1×, 2×, and 4× downsampling cases.

S3.3 CNN architecture

The input to the CNN has nine channels, corresponding to three stacked RGB inputs – the camera image whose height we wish to predict, followed by the left and right camera views (Fig. 2c). The output of the CNN is a single-channel height map, obtained by summing across the channel dimension of the final convolutional layer.

The encoder-decoder CNN architectures were based on one basic building block, consisting of the following operations in sequence:

1. 3×3 convolution, k filters, stride=1, padding='same',
2. Batch normalization,
3. Leaky ReLU,
4. 1×1 convolution, k filters, stride=1, padding='valid',
5. Batch normalization,
6. Leaky ReLU (unless final block of the CNN),

where k is a free hyperparameter, specifying the number of filters in the convolution layers. In the case of an upsample block, a 2× nearest-neighbor upsampling procedure is applied *before* the block. In the case of a downsample block, a 2×2 max pooling operation is applied *after* the block.

The full, symmetric encoder-decoder CNN architecture is described by a list of positive integers, each of which specifies the k for an upsample/downsample block pair. For example, [8, 16, 32] indicates three downsample blocks with k = 8, 16, and 32 filters, followed by three upsample blocks with k = 32, 16, and 8 filters. In our experiments, we set $k = 32$ for all upsample/downsample blocks, but varied the number of blocks between 3 and 6 (i.e., [32, 32, 32] and [32, 32, 32, 32, 32, 32]), depending on the sensor downsampling.

S3.4 Data-dependent loss function

The data-dependent loss function is computed based on the model depicted in Fig. 2a, where 2-3 image patches are used instead of 54 full-size images. Specifically, the 4-channel (RGBH) image patches are backprojected onto a blank

"canvas" according to the camera poses and height map-derived orthorectification fields (Eq. 1). The same coordinates are then used to reproject back to camera-centric coordinates to obtain the forward predictions. The data-dependent loss function is thus the MSE between forward predictions and the original RGBH patches.

S3.5 Normalized high-pass filtering

For terrestrial samples, which were illuminated in reflection, we found that registering the RGB images sometimes led to artifacts due to camera-dependent photometric appearance. This can be caused by illumination variation across the FOV due to off-axis LED panel geometry and anisotropic, non-Lambertian reflections, causing different amounts of light entering each camera. To combat these effects, we used normalized high-pass filtered versions of the images,

$$\widetilde{I}_{\sigma}(x,y) = \frac{I(x,y) \circledast \exp\left(-\frac{x^2+y^2}{4\sigma^2}\right)}{I(x,y) \circledast \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)}, \tag{S2}$$

where $\circledast$ denotes 2D convolution. Thus, Eq. S2 is the ratio of two Gaussian-blurred versions of $I(x,y)$, the grayscale-converted RGB image, with widths σ and $\sqrt{2}\sigma$. Like high-pass filtering, applying Eq. S2 to the images highlights edges and attenuates DC and low-frequency features. The motivation for taking a ratio rather than subtracting (i.e., difference of Gaussians) is so that the spatial fluctuations are normalized and therefore illumination-variation-independent, thereby facilitating registration. To capture different scales, we used three values of σ for the three image channels ($\sigma = 1, 2, 4$).

S3.6 Regularization of the height maps

In addition to the CNN reparameterization (i.e., DIP) of the height maps as a regularizer [1, 3, 4], we also incorporated two additive regularization terms to the overall loss function: height map consistency regularization and support regularization. The height map consistency regularization enforces agreement in height values in overlapped regions of camera images and simply comes from the fourth channel of the RGBH images, whose contribution can be scaled by a hyperparameter, λ_{height}. We observed smoothing effects with increasing λ_{height}. The object support regularization relies on a segmentation mask of the background pixels, whose height values we enforce to be a particular constant (e.g., 0) via an L2 loss. In other words,

$$loss_{support} = \lambda_{support} \sum_{x,y} mask_{background}(x,y)(h(x,y) - h_0)^2, \tag{S3}$$

where $mask_{background}(x,y)$ is the segmentation mask, $h(x,y)$ is the height map output of the CNN, h_0 is the known background height value, and λ_{height} is

the regularization coefficient. In this paper, we used a simple intensity-based threshold on the green channel of the photometric images, as our backgrounds are relatively homogeneous, although other segmentation strategies may be used.

S3.7 Additional training details

We optimized the loss function, consisting of the aforementioned data-dependent and regularization terms, using the Adam Optimizer [5]. Depending on the downsampling configuration, we used a different patch size and number of patches per iteration: one 1024×1024 patch (no downsampling), two 768×768 patches (2× downsampling), and eight 384×384 patches (4× downsampling). These patches were randomly selected from a subset of the recorded video frames – for the 2× and 4× downsampling configurations, we selected from 15-16 frames evenly distributed frames, while for the no downsampling configuration, we used 8 frames (due to memory constraints).

For the reflection-illuminated terrestrial samples, we performed a two-step training procedure, where we first optimized with RGB images using $\lambda_{height} = 500$ (Supplementary Sec. S3.6) to scale the height channel (with units of mm) and $\lambda_{support} = 0$ (Eq. S3) for 30k iterations. Thereafter, we ran 70k iterations with the normalized high-pass filtering (Supplementary Sec. S3.5) and $\lambda_{height} = 50$ and $\lambda_{support} = 100$. For aquatic samples, high-pass filtering was not necessary because they were illuminated in transmission. Thus, we used a one-step training procedure with 70k iterations with $\lambda_{height} = 50$ and $\lambda_{support} = 100$.

S3.8 Inference step - generating the full-size RGBH videos

Once the CNN is trained to map from multi-ocular stereo inputs to a 3D height map using the patch-based procedure, we can apply the CNN to sequences of full-sized MCAM video streams that includes unseen frames (Fig. S4). Essentially, this refers to the backprojection step in Fig. 2a. Since iterative optimization is no longer necessary after the CNN is fully trained, generating new 3D video frames can be done quickly. For example, one application might involve a human observer selecting a particular region of interest within the large FOV, whose 3D height map the computer would then generate in real time.

S4 Reducing the impact of the per-camera rolling shutter

Each sensor exhibits a rolling shutter, whereby the pixels begin integrating sequentially every $\delta t = (230\ \text{MHz})^{-1} = 4.35$ ns and are read out in a raster scan pattern row by row from the top left to bottom right (with the longer sensor dimension as the horizontal dimension). Although the rolling shutters

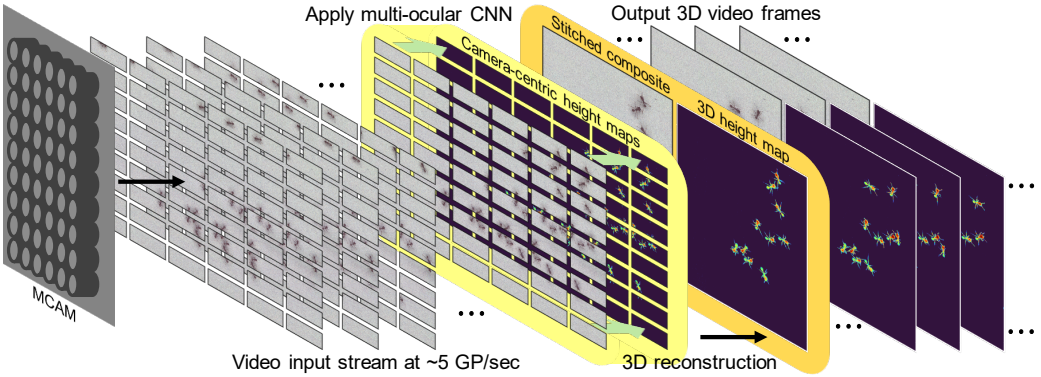

Fig. S4 Inference step post patch-based training (Fig. 2c) that generates the stitched composites and coregistered 3D height map on potentially unseen video frames.

are synchronized to within 10 µs across cameras, there is still significant asynchrony in overlapped regions of neighboring camera FOVs, thus thwarting accurate 3D estimation. Here, we consider asynchrony in 1) vertically overlapped FOVs and 2) horizontally overlapped FOVs. The former asynchrony is much more serious, as the bottom row of the upper sensor is not reached until after $\delta t \times l_{row} \times l_{col}$, where l_{row} and l_{col} are the number of pixels per row and column, respectively. Using the full sensor without downsampling ($l_{row} = 4208$, $l_{row} = 3120$), the time delay between the last row of the upper sensor and the first row of the lower sensor is ∼57 ms. In practice, the delay is even larger due to horizontal and vertical blanking (dead time between row and column reads). To circumvent this problem, we thus reduced the number of rows approximately in half (3120 to 1536) to ensure the smallest overlap between vertically adjacent cameras that still allowed for a contiguous composite FOV. This also has the added benefit of increasing the sensor frame rate.

Asychrony in horizontally overlapped FOVs is less serious, but still an important consideration. Using the full sensor without downsampling, the time delay between corresponding rows of perfectly aligned camera FOVs is only $\delta t \times l_{row}$, or approximately 20 µs, which is negligible. In practice, however, there is a vertical offset due to slight camera misalignments, so that the time delay is $\delta t \times l_{row} \times l_{misalign}$. Based on stitching a flat target, we determined that the worst-case vertical misalignment was $l_{misalign} = 100$ rows, leading to a 2-ms delay between when the corresponding pixels in horizontally neighboring cameras begin to expose. To ensure significant temporal overlap (at least 90%) in the exposure periods, we thus exposed for 2 ms$/(1 - 0.9) = 20$ ms.

For 2× and 4× downsampling, the asynchrony is less dramatic because the numbers of rows and columns are reduced. Going through similar calculations, we determined that exposing for 5 ms and 2.5 ms for 2× and 4× downsampling, respectively, leads to >90% temporal overlap in the worst-case vertical camera misalignment cases. Note that these values don't quite scale proportionally

between the 2× and 4× cases due to horizontal blanking periods not decreasing proportionally.

S5 Impact of hardware design on height accuracy

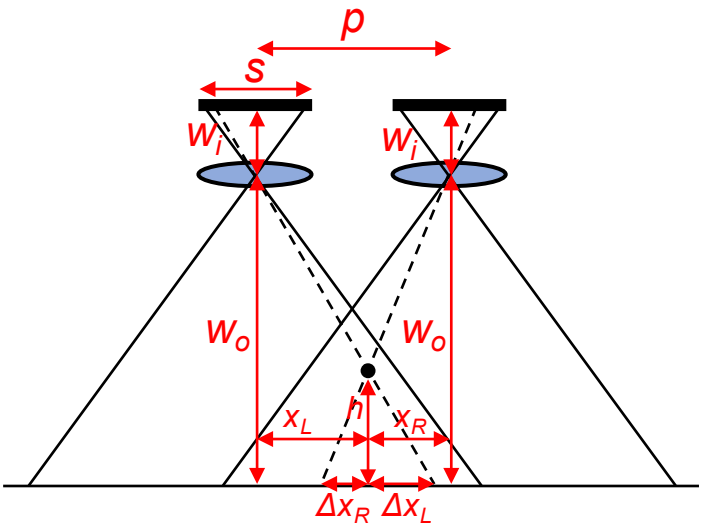

Fig. S5 Two identical cameras with effective focal length f observing a common sample point with height h from the focal plane. The magnification is $M = w_i/w_o$.

Here, we explore how hardware design choices impact the accuracy of 3D height estimation. We will ignore errors stemming from camera distortion, aberrations, and misalignment and assume ideal paraxial imaging performance. Further, for simplicity, we assume two adjacent cameras spaced by p center to center with a common effective focal length, f, a working distance (i.e., the distance between the sample plane and the lens principal plane) of w_o, and a sensor-to-lens distance of w_i (Fig. S5). These latter three parameters satisfy the lens equation,

$$\frac{1}{w_o} + \frac{1}{w_i} = \frac{1}{f}. \tag{S4}$$

The magnification is thus $M = w_i/w_o$.

Further, consider a sample point with height h positioned x_L from the optical axis of the left camera and x_R from that of the right camera. Due to nontelecentric optics, the apparent object-side position of this sample point is parallax-shifted Δx_L in the left camera and Δx_R in the right camera. These

shifts are related to the height via Eq. 1,

$$\Delta x_L = \frac{h x_L M}{f(M+1) - hM}, \quad \Delta x_R = \frac{h x_R M}{f(M+1) - hM}. \tag{S5}$$

We are interested in the total parallax shift between both cameras, given by

$$\Delta x = \Delta x_L + \Delta x_R = \frac{hpM}{f(M+1) - hM}, \tag{S6}$$

which does not depend on the lateral position of the sample point, as $x_L + x_R = p$. How well we can estimate Δx depends on how accurately we can match and register the sample point in both camera images, which in turn depends on the lateral resolution of the imaging system. We consider two limits: the diffraction-limited regime and the pixel-size-limited regime. Let δx_{pixel} be the camera pixel size, so that $\delta x_{pixel}/M$ is the object-side pixel size. Further, let δx_{diff} be the camera-side diffraction-limited spot size, so that $\delta x_{diff}/M$ is the object-side diffraction-limited spot size:

$$\delta x_{diff} \propto \frac{\lambda}{NA} \approx \frac{2\lambda w_i}{w} = \frac{2\lambda f(M+1)}{w}, \tag{S7}$$

where w is the lens aperture diameter and λ is the wavelength. Assuming that we can match corresponding points in the two camera images with an uncertainty proportional to the lateral resolution, then the corresponding height error can be estimated by setting Δx (Eq. S6) equal to the object-side lateral spot size and solving for h. In the pixel-resolution-limited regime ($\delta x_{pixel} \gg \delta x_{diff}$), we have that the height uncertainty is

$$\delta h_{pixel} \propto \frac{f \delta x_{pixel}(M+1)}{M(\delta x_{pixel} + pM)}, \tag{S8}$$

meaning that downsampling the images results in a roughly proportional decrease in height uncertainty. In the diffraction-limited regime, we have that

$$\delta h_{diff} \propto \frac{2\lambda f^2 (M+1)^2}{M(2\lambda f(M+1) + pwM)}. \tag{S9}$$

We can see that in both cases, all else equal, decreasing f and increasing p and M improve the height estimation accuracy. It may appear helpful to decrease M to increase the amount of overlap of neighboring camera FOVs until eventually non-adjacent cameras begin to overlap, resulting in larger values of p. However, in both pixel-limited and diffraction-limited regimes, $1/p$ decreases more slowly than the factors that include M increase as M decreases (e.g., consider $p \to 2p$, $M \to M/2$). Furthermore, this analysis assumes that the object height variation is within the depth of field of the imaging systems, within which the lateral resolution remains roughly constant. Thus, while designs that

increase the lateral resolution can improve height estimation accuracy, they also compromise the axial FOV.

We now consider the case where the camera FOVs are critically overlapped at 50%, that is when $M = s/2p$, where s is the sensor width. Thus, the height uncertainties in the pixel- and diffraction-limited regimes are, respectively,

$$\delta h_{pixel} \propto \frac{2\delta x_{pixel} f(2p+s)}{s(2\delta x_{pixel}+s)} \approx \frac{2\delta x_{pixel} f(2p+s)}{s^2}, \tag{S10}$$

$$\delta h_{diff} \propto \frac{2\lambda f^2(2p+s)^2}{s(2\lambda f(2p+s)+psw)} \approx \frac{2\lambda f^2(2p+s)^2}{ps^2w}. \tag{S11}$$

In the ideal case of $p = s$, so that there are no gaps in between the sensors and $M = 1/2$, we have

$$\delta h_{pixel} \propto \frac{\delta x_{pixel} f}{p}, \tag{S12}$$

$$\delta h_{diff} \propto \frac{\lambda f^2}{pw}. \tag{S13}$$

S6 SNR considerations

As with all imaging systems, SNR is an important metric for 3D-RAPID. Specifically, the better the SNR of the photometric images, the higher the image registration accuracy and by extension the 3D estimation accuracy. There are several trade offs involving SNR with our method as it relates to imaging small model organisms.

1. Numerical aperture (NA): the higher the NA, the more light collected and the better the shot-noise-limited SNR. The associated improved lateral resolution also improves the 3D height estimation accuracy, because the parallax estimation accuracy would increase (Supplementary Sec. S5). However, at the same time, the higher the NA, the shallower the depth of field, which limits the axial FOV of the 3D reconstructions. In addition, the higher the NA, the smaller the lateral FOV becomes in practice due to difficulties in correcting aberrations [6] and therefore the tighter the camera array packing would need to be.
2. Behavior: while increasing the illumination power would yield higher SNR, care must be taken to avoid influencing the behavior of the model organisms. This tradeoff can be partially alleviated by using wavelengths invisible to the model organism's visual system, however radiative heating from the illumination source can potentially still influence behavior.
3. Speed: the higher the frame rate, the less light that is detected and therefore the lower the SNR per frame. Increasing illumination power can alleviate this tradeoff until it influences the behavior of interest.
4. Camera type: one of the factors enabling the financial tractability of the 3D-RAPID architecture is its use of CMOS digital image sensors that are currently fabricated at large scales for the cell phone camera market. While

the sensitivities of these camera sensors have improved significantly over the past decade (e.g., now with very low read noise and dark current and high quantum efficiency, due in part to the introduction of back-side illuminated CMOS sensors), their performance may still generally lag behind that of high-end scientific CMOS and EMCCD sensors. While this latter technology is currently too expensive to multiplex into an array with more than several dozen sensors, it may become feasible in the future.

S7 Supplementary video descriptions

1. 60-fps, 36.6-MP video of freely swimming zebrafish larvae (10 dpf) feeding on mostly floating AP100 food particles. The left panel is the photometric composite and the right panel is the 3D height map. The video zooms into three feeding events (or attempts) by two different fish.
2. 230-fps, 9.1-MP video of freely swimming zebrafish larvae (10 dpf) feeding on mostly floating AP100 food particles. The left panel is the photometric composite and the right panel is the 3D height map. The video zooms in on three independent feeding events by three different fish. The third fish can be seen swallowing the food particle.
3. 60-fps, 36.6-MP video of freely swimming zebrafish larvae (10 dpf) feeding on mostly floating AP100 food particles. The left panel shows the full field of view with the trajectories mapped out. The panels on the right each correspond to individual fish, uniquely identified by a 2-digit number, whose position and orientation are denoted with red annotations. The righthand panels' border colors nonuniquely match those of the tracks in the lefthand panel, to assist the viewer in matching the fish to the trajectories. Righthand panels appear and disappear when the fish enters or exits the FOV. The first half of the video shows the photometric values, while the second half of the video shows the 3D height maps.
4. 60-fps, 36.6-MP video of 20-dpf zebrafish larvae feeding on live brine shrimp. The left panel is the photometric composite and the right panel is the 3D height map. The video zooms in on two feeding events from two different fish.
5. 230-fps, 9.1-MP video of 20-dpf zebrafish larvae feeding on live brine shrimp. The left panel is the photometric composite and the right panel is the 3D height map. The video zooms into one feeding event.
6. 60-fps, 36.6-MP video of a large school of 5-dpf zebrafish larvae freely swimming in an open arena at high speed. The left panel is the photometric composite and the right panel is the 3D height map.
7. 230-fps, 9.1-MP video of a large school of 5-dpf zebrafish larvae freely swimming in an open arena at high speed. The left panel is the photometric composite and the right panel is the 3D height map.
8. 60-fps, 36.6-MP video of freely moving fruit flies. The left panel is the photometric composite and the right panel is the 3D height map.

9. 230-fps, 9.1-MP video of freely moving fruit flies. The left panel is the photometric composite and the right panel is the 3D height map.
10. 60-fps, 36.6-MP video of freely moving fruit flies. The left panel shows the full field of view with the trajectories mapped out. The panels on the right each correspond to individual flies, uniquely identified by a 2-digit number, whose position is denoted by a red circle. The righthand panels' border colors nonuniquely match those of the tracks in the lefthand panel, to assist the viewer in matching the flies to the trajectories. Righthand panels appear and disappear when the fish enters or exits the FOV. The first half of the video shows the photometric values, while the second half of the video shows the 3D height maps.
11. 60-fps, 36.6-MP video of freely moving harvester ants. The left panel is the photometric composite and the right panel is the 3D height map.
12. 230-fps, 9.1-MP video of freely moving harvester ants. The left panel is the photometric composite and the right panel is the 3D height map.

References

[1] Zhou, K.C., Cooke, C., Park, J., Qian, R., Horstmeyer, R., Izatt, J.A., Farsiu, S.: Mesoscopic photogrammetry with an unstabilized phone camera. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7535–7545 (2021)

[2] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., *et al.*: {TensorFlow}: A system for {Large-Scale} machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283 (2016)

[3] Ulyanov, D., Vedaldi, A., Lempitsky, V.: Deep image prior. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9446–9454 (2018)

[4] Zhou, K.C., Horstmeyer, R.: Diffraction tomography with a deep image prior. Optics Express **28**(9), 12872–12896 (2020)

[5] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[6] Zheng, G., Ou, X., Horstmeyer, R., Chung, J., Yang, C.: Fourier ptychographic microscopy: A gigapixel superscope for biomedicine. Optics and Photonics News **25**(4), 26–33 (2014)