Focal Split: Untethered Snapshot Depth from Differential Defocus

Junjie Luo[*,1], John Mamish[*,2], Alan Fu[*,1], Thomas Concannon[1],
Josiah Hester[2], Emma Alexander[3,†], and Qi Guo[1,‡]
[1]Elmore Family School of Electrical and Computer Engineering, Purdue University
[2]College of Computing, Georgia Institute of Technology
[3]McCormick School of Engineering, Northwestern University
[*] Equal contributions, [†]ealexander@northwestern.edu, [‡]qiguo@purdue.edu

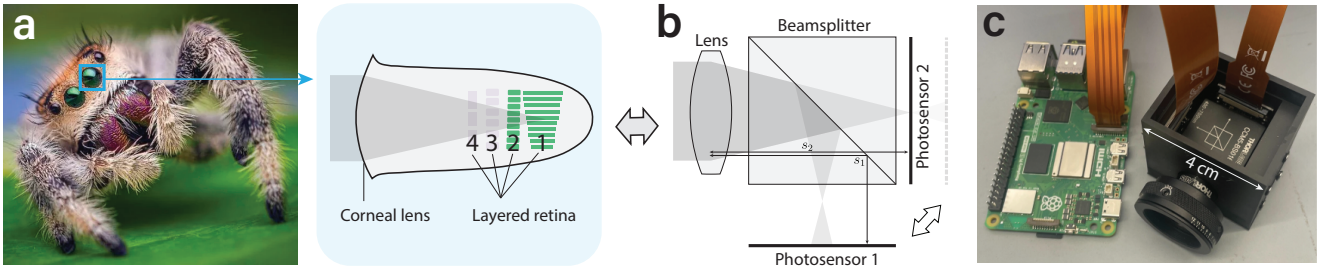

Figure 1. Overview. (a) The principal eyes of jumping spiders comprise layered retinae, allowing the same scene to be imaged simultaneously at slightly different distances from the lens. This enables them to see two differentially defocused images of a target, from which depth can be estimated efficiently [22]. (b) Focal Split's novel optomechanical setup leverages a beamsplitter and two photosensors placed at different sensor distances to the lens to mimic the jumping spider's eye structures. (c) Our handheld, untethered Focal Split prototype can generate real-time sparse depth maps from battery-powered on-board computing.

Abstract

We introduce Focal Split, a handheld, snapshot depth camera with fully onboard power and computing based on depth-from-differential-defocus (DfDD). Focal Split is passive, avoiding power consumption of light sources. Its achromatic optical system simultaneously forms two differentially defocused images of the scene, which can be independently captured using two photosensors in a snapshot. The data processing is based on the DfDD theory, which efficiently computes a depth and a confidence value for each pixel with only 500 floating point operations (FLOPs) per pixel from the camera measurements. We demonstrate a Focal Split prototype, which comprises a handheld custom camera system connected to a Raspberry Pi 5 for real-time data processing. The system consumes 4.9 W and is powered on a 5 V, 10,000 mAh battery. The prototype can measure objects with distances from 0.4 m to 1.2 m, outputting 480×360 sparse depth maps at 2.1 frames per second (FPS) using unoptimized Python scripts. Focal Split is DIY friendly. A comprehensive guide to building your own Focal Split depth camera, code, and additional data can be found at https://focal-split.qiguo.org.

1. Introduction

Depth from differential defocus (DfDD) is a family of physically rigorous depth-sensing methods that generate a depth map from a series of *differentially defocused* images with extremely efficient computations [1]. Its development was partially inspired by the optical functionalities of jumping spider's eyes [16, 22]. Here, we address two major shortcomings in this family. First, existing DfDD cameras either require sequential image captures and assume the scene to be static [7, 20], additional computation to handle motion between frames [2, 3], or narrowed illumination bandwidth in conjunction with a multifunctional metasurface [8]. In contrast to the snapshot multi-focus capture of the spider's layered retina (Fig. 1a), this limits their accuracy or light efficiency in the face of moving scenes under normal illumination. Second, despite the promise of DfDD algorithms for low-power applications, these prototypes have all been tethered to wall sockets for power [7] and to laptop computers for depth map calculation [2, 7, 8, 20].

We present *Focal Split*, the first snapshot, untethered DfDD camera. As shown in Fig. 1c, Focal Split utilizes a novel optical design that functionally mimics jumping spiders' eye structure. It splits the incident light focused by a

lens through a beamsplitter to form two images with different sensor distances s_1 and s_2. By synchronizing the two photosensors, the hardware captures a pair of images, I_1 and I_2, with varied sensor distances simultaneously, effectively resembling a pair of layered semi-transparent retinae. Compared to previous DfDD sensors that sequentially capture the differently defocused images by varying the focal length of the lens [7], aperture diameter [28], aperture code [34], or camera positions [3], Focal Split achieves the critical advantage of capturing I_1 and I_2 in a snapshot. This avoids misalignment between the images when calculating the depth map, which, as we will show, seriously contaminates the quality of the depth maps.

We also derive a new physically rigorous depth estimation algorithm specialized for the optical setup of this paper. It shows that the depth map of the scene can be calculated by a simple pixel-wise expression with the image derivatives:

$$Z = \frac{a}{b + I_s/\nabla^2 I}, \tag{1}$$

where a and b are constants determined by the optics and I_s and $\nabla^2 I$ are the image derivatives that can be estimated from the pair of differently defocused images, I_1 and I_2, after aligning their magnification. The proposed algorithm is a novel instance that belongs to the family of depth from differential defocus [1]. In this paper, we perform comprehensive theoretical and simulation analyses on the proposed algorithm's sensitivity to texture frequency, noise, etc.

Our algorithm can only produce a partially dense depth map from the input images. This is because Eq. 1 degenerates at textureless regions of the image (I_s and $\nabla^2 I$ becomes zero.) In fact, all passive-ranging methods, including stereo, fundamentally fail at textureless regions. Previous algorithms typically perform passive ranging and implicit depth map densification in a single model. These methods cost relatively high computation but produce high-quality, dense depth maps. Compared to them, our algorithm provides an alternative option to generate a partially dense depth map with a much lower computation, leaving the densification to downstream tasks, which is suitable for low-power, autonomous platforms, such as micro-robots, autonomous underwater vehicles, AR glasses, etc. Using Eq. 1, Focal Split only costs 500 floating point operations (FLOPs) per pixel to generate a partially dense depth map.

Focal Split is the first untethered DfDD camera. As shown in Fig. 1c, this handheld system comprises a custom housing of the optics and photosensors and a Raspberry Pi 5 that performs real-time onboard depth estimation. The housing is 4 cm $\times$ 5 cm $\times$ 6 cm. The system can output depth maps at 480$\times$360 resolution at 2.1 FPS using unoptimized Python scripts with the power consumption of only 4.9 W. The contributions of this work can be summarized as follows:

1. **A new optical design** that functionally mimics the principle eye of jumping spiders. It enables the simultaneous capture of a pair of differentially-defocused images.
2. **A new DfDD algorithm** with a verified advantage in robustness for our optical design.
3. **A new low-power, untethered working prototype for snapshot depth sensing** with a power budget < 10 W and a significant error reduction compared to sequential measurement DfDD for dynamic scenes.

2. Related Work

Techniques for depth imaging can broadly be divided into two categories: active and passive. Generally speaking, techniques from each of these categories have somewhat similar characteristics; active imaging systems typically have worse size, weight, power, and cost (SWaP-C) but better accuracy, while passive imaging systems being complementary with better SWaP-C and worse accuracy [5]. Below, we will give a discussion of power consumption and portability for depth imaging systems using both active and passive techniques.

2.1. Active techniques

"Active imaging" is a descriptor for any system which must project light into a scene in order to image it. Because of their accuracy, active imaging systems are considered to be the gold standard for depth imaging. However, due to the power needed to illuminate scenes, active imagers are not appropriate for low-power, portable applications [5].

LiDAR. LiDAR systems are active imaging systems which work by sending laser pulses into a scene and recording their round-trip time-of-flight with high precision to calculate distance. However, high-powered lasers are required for LiDAR systems to operate, limiting their portability [5]. Furthermore, LiDAR systems require their light source to be scanned across a scene, meaning that bulky and power-consuming opto-mechanical setups must be used [27]. Some recent works improve the power consumption of LiDAR systems by introducing novel scanning mechanisms [25] and deep-learning based methods for depth completion on sparse maps [4, 31], but these methods depend on novel opto-mechanical components and large DNNs, limiting their adoption and restricting their use in edge systems.

Structured Light. Structured light 3D scanners operate by illuminating a scene with a specifically chosen pattern and imaging the resulting scene. Because the projected light pattern is known, the measured points can be used to calculate depth [6, 12]. Structured light systems require less power than LiDAR systems [12] because they do not require nanosecond-level timing precision, but they are less robust and require sophisticated image processing pipelines to calculate depth images [6].

2.2. Passive techniques

Unlike active imaging systems, passive depth imagers do not emit any light. This removes the most energy intensive component of active systems, but passive depth imagers often require more sophisticated image processing pipelines and are less robust [14].

Monocular Structure from Motion (SfM). Structure from motion (SfM) refers to the recovery of 3D structure from a sequence of monocular images [24, 26]. Although SfM techniques are appealing because their monocular image sensors are compact and cheap [24], viable SfM methods are computationally intensive and require significant memory, restricting their adoption in portable, low-power, real-time systems [13, 21, 29].

Stereo vision. Stereo vision is one of the most well-researched methods for producing depth images from conventional image sensors. By positioning 2 cameras a fixed distance apart, depth can be recovered by looking at the differences in matching scene features' positions between the images. While conceptually simple, this problem is ill-posed under many circumstances, with many inverse solutions [15]. In the literature, this problem is referred to as *stereo matching* and has been researched for decades [10, 15, 17, 33].

Although stereo vision systems can deliver quality depth maps, robust stereo matching searches the left and right images globally for matches, making them computationally expensive [10, 33]. By restricting the range of stereo matching searches, more energy-efficent stereo methods can be developed. Some systems have even been developed which use local matching on FPGAs or ASICs. Despite significant advances [19, 26], power/performance trade-offs remain challenging for fully-realized real-time systems.

Recently, deep learning has been explored as an alternative to classical search-based stereo matching methods. Although these methods achieve very promising performance when compared with classical stereo methods, they require at least 100s of MB of RAM and GOPS of compute, ruling them out for low-power edge applications [15].

2.3. Depth from Defocus

Depth from Defocus (DfD) is a passive method for generating depth maps from images by analyzing the defocus blurs. This class of method traditionally requires specialized optical modulation in the imaging formation process for engineered defocus, such as phase or amplitude aperture masks [9, 18, 34]. Recently, learning-based DfD algorithms have demonstrated the capability to generate high quality depth maps in real time [9, 11, 32]. DfD can be performed using a single camera, an advantage in spatial compactness compared to stereo.

Recently, depth from differential defocus (DfDD) has emerged as a computationally efficient alternative to traditional DfD methods. Analogous to the computational savings in optic flow from using differential brightness constancy in place of feature tracking, DfDD produces depth maps efficiently by solving closed-form equations on image derivatives caused by differential changes in defocus. DfDD prototypes run at up to 100 FPS, but have still relied on powerful workstations to accomplish this performance [7].

Additionally, previous DfDD cameras have handled scene motion in a variety of ways. While early work used motion as its defocus cue [2, 3], this method proved less stable and more computationally expensive than optically-controlled defocus. Tunable-lens-based DFDD [7, 20] proved more effective, but due to time-multiplexing the focus changes, are vulnerable to scene motion within paired frames. A metalens-based snapshot camera addresses this issue, but requires custom-fabricated hardware and a narrow bandwidth illumination [8], limiting its application in practice.

3. Methods

As detailed in Section 2.3, DfDD is a passive and computationally efficient depth imaging technique, making it attractive for enabling low-power, untethered depth cameras. However, prior DfDD work uses time-division multiplexing to capture differently focused images using the same sensor. This performs poorly under scenes with motion, making it unsuitable for dynamic scenes and cameras that experience ego-motion.

Focal Split overcomes this issue by introducing a second image sensor. This allows two differently focused images of the same scene to be captured simultaneously. Our contributions include the optomechanical (Section 3.1) and mathematical (Section 3.2, 3.3) developments required to accommodate a second image sensor.

3.1. Optomechanical Design

As shown in Fig. 1b, Focal Split captures the target scene through a single lens and uses a beamsplitter to guide image copies to two separate sensors, where they are digitized. In order to perform DfDD, the captured images must somehow differ in their focus, which Focal Split achieves by placing sensors at different optical distances.

Unlike [8], Focal Split's achromatic optomechanics work across the visible spectrum and can be constructed at low-cost with a 3D-printed enclosure and commodity off-the-shelf components, making the creation of Focal Split-based systems widely accessible.

3.2. Depth from Differential Defocus

As shown in Fig. 2, consider a simple scenario in which a front-parallel plane placed at distance Z with texture

(spatially-varying brightness) T is imaged through an ideal thin lens with Gaussian blur. Additionally, the distance s between the sensor and the lens, the sensor distance, is allowed to vary, forming an image $I(\boldsymbol{x}; s)$. According to the thin-lens model, the image formation process can be described mathematically with a convolution in $\boldsymbol{x}$,

$$I(\boldsymbol{x}; s) = k(\boldsymbol{x}; s) * P(\boldsymbol{x}; s), \tag{2}$$

between the all-in-focus pinhole image $P(\boldsymbol{x})$:

$$P(\boldsymbol{x}; s) = T\left(-\frac{Z}{s}\boldsymbol{x}\right), \tag{3}$$

and the Gaussian PSF $k(\boldsymbol{x})$:

$$k(\boldsymbol{x}; s) = \frac{1}{\sigma^2}\exp\left(\frac{\|\boldsymbol{x}\|^2}{2\sigma^2}\right), \tag{4}$$

where the standard deviation is the defocus level σ:

$$\sigma = A\left(\frac{1}{Z} - \rho\right)s + A. \tag{5}$$

The defocus level σ is a function of the object distance Z, the optical power ρ, the lens-to-sensor distance s, and the standard deviation of the Gaussian aperture code A.

We note that both the pinhole image $P(\boldsymbol{x}; s)$ and the PSF $k(\boldsymbol{x}; s)$ depend on the sensor distance s. Previous methods account for pinhole magnification change with additional image derivatives (Fig. 2 blue). We improve robustness and efficiency by correcting the magnification change to isolate the defocus effect. To do this, we register both images to a consensus sensor location c. The aligned image $\tilde{I}$ is generated from a measurement $I(\boldsymbol{x}; s)$ by a simple spatial scaling:

$$\tilde{I}(\boldsymbol{x}; s) = I\left(\frac{s}{c}\boldsymbol{x}\right). \tag{6}$$

The scaled image $\tilde{I}(\boldsymbol{x}; s)$ can be expressed as the convolution of the consensus pinhole image $P(\boldsymbol{x}; c)$, which is independent of s, and a scaled PSF $\tilde{k}(\boldsymbol{x}; s)$:

$$\tilde{I}(\boldsymbol{x}; s) = \tilde{k}(\boldsymbol{x}; s) * P(\boldsymbol{x}; c). \tag{7}$$

The scaled PSF $\tilde{k}(\boldsymbol{x}; s)$ has the form:

$$\tilde{k}(\boldsymbol{x}; s) = \left(\frac{s}{c}\right)^2 k\left(\frac{s}{c}\boldsymbol{x}\right), \tag{8}$$

which has the following mathematical relationship between its derivatives:

$$\tilde{k}_s(\boldsymbol{x}; s) = -\frac{c^2\sigma A}{s^3}\nabla^2\tilde{k}(\boldsymbol{x}; s), \tag{9}$$

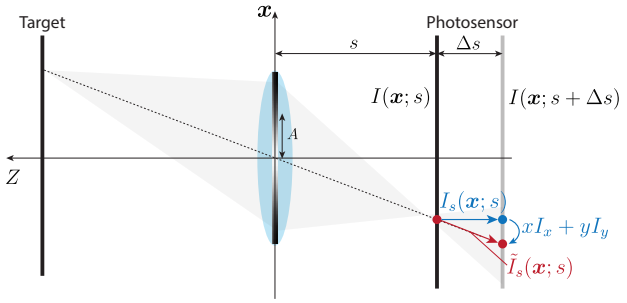

Figure 2. The image formation model. The proposed algorithm calculates the derivative of the aligned images, $\tilde{I}_s$, as a cue for object depth (red arrow). In contrast, previous work [1] uses two derivatives (blue arrows), I_s and $xI_x + yI_y$, to approximate the same quantity, resulting in higher computation and numerical instability.

where ∇^2 is the Laplacian in $\boldsymbol{x}$. Because the consensus pinhole image is independent of s, the scaled image $\tilde{I}(\boldsymbol{x}; s)$ follows a similar relationship:

$$\begin{aligned}\tilde{I}_s(\boldsymbol{x}; s) &= \tilde{k}_s(\boldsymbol{x}; s) * P(\boldsymbol{x}; c) \\ &= -\frac{c^2\sigma A}{s^3}\nabla^2\tilde{k}(\boldsymbol{x}; s) * P(\boldsymbol{x}; c) \\ &= -\frac{c^2\sigma A}{s^3}\nabla^2\tilde{I}(\boldsymbol{x}; s).\end{aligned} \tag{10}$$

Using Eq. 5 for σ, we obtain the following equation for scene distance Z, which applies at every pixel $\boldsymbol{x}$:

$$Z(\boldsymbol{x}) = \frac{a}{b + \tilde{I}_s(\boldsymbol{x}; s)/\nabla^2\tilde{I}(\boldsymbol{x}; s)}, \tag{11}$$

where a and b are optical constants:

$$a = -A^2,\ b = -A^2(1/f - 1/s).$$

For simplicity of notation, we will drop the $(\boldsymbol{x})$ from the equations hereafter whenever the calculation is per-pixel. The full derivation is provided in the supplementary.

Eq. 11 can be implemented using the proposed optomechanical setup shown in Fig. 1c. By capturing a pair of images I_1 and I_2 at different sensor distances $I_1(\boldsymbol{x}) = I(\boldsymbol{x}; s_1)$ and $I_2(\boldsymbol{x}) = I(\boldsymbol{x}; s_2)$, we approximate the image derivatives via:

$$\begin{aligned}\tilde{I}_{s,\text{approx}} &= I_1(R\boldsymbol{x} + \boldsymbol{t}) - I_2(\boldsymbol{x}), \\ \nabla^2\tilde{I}_{\text{approx}} &= \frac{1}{2}\nabla^2\left(I_1(R\boldsymbol{x} + \boldsymbol{t}) + I_2(\boldsymbol{x})\right),\end{aligned} \tag{12}$$

where the matrix $R \in \mathbb{R}^{2\times2}$ and vector $\boldsymbol{t} \in \mathbb{R}^{2\times1}$ describe a homography that aligns the images I_1 and I_2, including the rescaling.

3.3. Confidence and sensitivity analysis

Confidence. Like other DfDD algorithms, our new depth equation (Eq. 11) generates a solution even in textureless or blurred-out regions, but the lack of image contrast will cause numerical instability in the ratio of small derivatives $\tilde{I}_s$ and $\nabla^2\tilde{I}$. Fortunately, this degeneracy can be predicted with confidence from the measured values of $\tilde{I}_s$. As shown from the simulation result in Fig. 3, the overall depth estimation error is approximately inversely proportional to the magnitude of the image derivative $|\tilde{I}_s|$. Thus, we can define a simple confidence metric at each pixel:

$$C = \tilde{I}_s^2, \tag{13}$$

and use the confidence value C to filter out depth predictions according to a preset threshold C_{thre}. Sec. 5 shows the effectiveness of the confidence metric in the real data.

Working range. As objects become farther away from the plane of focus, the defocus blur gradually attenuates the intensity variations in the images. Meanwhile, as the image noise stays constant, the signal-to-noise ratio (SNR) of both image derivatives $\tilde{I}_s$ and $\nabla^2\tilde{I}$ gradually reduces. Here, we mathematically define the SNR of $\tilde{I}_s$ and $\nabla^2\tilde{I}$ as:

$$\begin{aligned} \text{SNR}(\tilde{I}_s) &= \frac{\tilde{I}_s}{|\tilde{I}_s - \tilde{I}_{s,\text{approx}}|}, \\ \text{SNR}(\nabla^2\tilde{I}) &= \frac{\tilde{I}_s}{|\nabla^2\tilde{I} - \nabla^2\tilde{I}_{\text{approx}}|}, \end{aligned} \tag{14}$$

where the subscript approx indicate the approximated derivatives using Eq. 12. Fig. 3b validates the significant decrease in SNR of $\tilde{I}_s$ and $\nabla^2\tilde{I}$ as the object departs from the plane of focus (dashed line.)

This evidence suggests that the proposed method has a natural *working range*, i.e., a region around the plane of focus where the algorithm's prediction is accurate. Empirically, we determine the working range as the depth region with a mean depth prediction error smaller than 5% of the true depth.

Numerical accuracy. The new depth equation (Eq. 11) outperforms the previously suggested equation for a layered-retina system [1], where Z was directly calculated from derivatives of the un-aligned images $I(\boldsymbol{x}; s)$ via:

$$Z = \frac{a}{b + \left(d(xI_x + yI_y) + I_s\right)/\nabla^2 I}, \tag{15}$$

with the additional constant $d = 1/s^2$ on a magnification term $xI_x + yI_y$ that our method does not need to compute. As illustrated in Fig. 2, the two methods, Eq. 11 and Eq. 15, differ in how they account for brightness changes across sensor distances. By rescaling the images, the proposed method removes the effect of the magnification term $xI_x + yI_y$ and loosens the requirement that the sensor distance change is differential in scale. The proposed method (Eq. 11) is more robust to larger variations in magnification than Eq. 15. This can be shown by the simulation analysis in Fig. 3c, where we study the depth accuracy as a function of sensor distance variation, $\Delta s = |s_1 - s_2|$. In the absence of noise, both methods show a rise in depth prediction error when Δs increase, but the proposed algorithm (Eq. 11) consistently achieves a lower error than the prior one (Eq. 15).

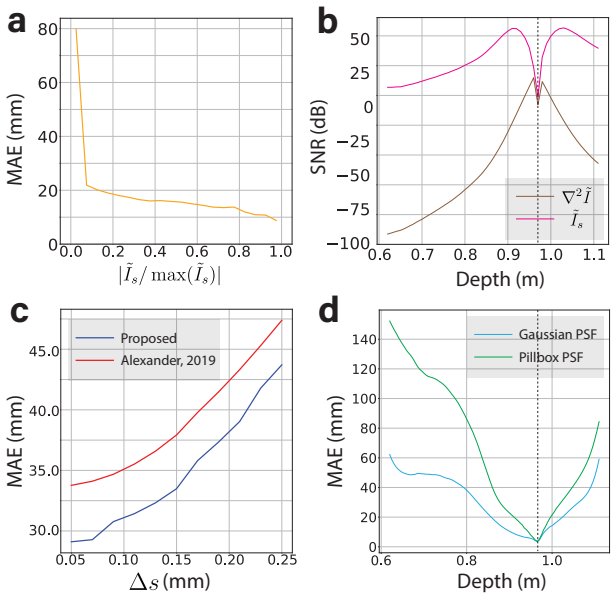

Figure 3. Sensitivity analysis using synthetic data. We simulate the image pair, I_1 and I_2, of front parallel textured planes placed at different depths Z and use the data to analyze the sensitivity of the algorithm. (a) Validation of the confidence metric. The overall depth estimation error, quantified by the mean absolute error (MAE), monotonically decreases as the normalized image derivative $|\tilde{I}_s/\max(\tilde{I}_s)|$ increases, suggesting the latter to be an effective confidence metric of the depth prediction. (b) Signal-to-noise ratio (SNR) of the estimated image derivatives $\tilde{I}_s$ and $\nabla^2\tilde{I}$ from finite difference. The vertical dashed line indicates the depth of the focal plane. The depth estimation becomes noisy when the SNR of both derivatives is too low. (c) Overall depth estimation error of using the proposed depth equation (Eq. 11) vs. the previously suggested equation (Eq. 15). The proposed one universally achieves higher accuracy for all sensor distance variation Δs. (d) Depth estimation error for Gaussian and Pillbox-shaped PSFs.

Aperture code. The derivation of Eq. 11 requires Gaussian PSFs. However, in practice, most cameras have a disk aperture, which leads to pillbox-shaped PSFs. Fig. 3d shows an increase in depth prediction error if directly using images captured with pillbox PSFs in simulation. However, it indicates the proposed algorithm can be directly applied to pillbox PSFs if the error can be tolerated.

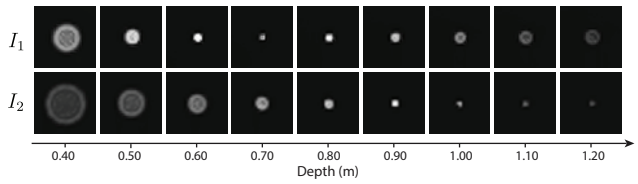

Figure 4. PSFs of the image pair, I_1 and I_2, at different depths using the assembled Focal Split prototype. The PSFs are measured by taking pictures of a white LED point source. The focal planes of I_1 and I_2 are approximately at 0.7 m and 1.2 m, respectively.

4. Prototype System

Utilizing the proposed depth sensing algorithm (Eq. 11), we design and build a depth camera with fully onboard power and compute using off-the-shelf optics, image sensors, and a single-board computer housed in a custom 3D-printed enclosure. The system can be powered by any 5VDC power bank, making it untethered and handheld. We include a complete DIY guide in the supplementary with a list of parts, the 3D-printing model, and assembly and calibration instructions to rebuild the prototype with $500 budget. As a high-level summary, the system (Fig. 1c) consists of a 30 mm lens, a non-polarizing cube beamsplitter, and two OV5647 RGB image sensors connected to a Raspberry Pi 5 to perform image capture and data processing. The two photosensors are designed to have a 0.4 mm difference in sensor distances.

4.1. Implementation

After simultaneously capturing the two images, I_1^{RGB} and I_2^{RGB}, from the two photosensors. We convert them to grayscale images, I_1 and I_2, for the depth estimation.

Aberration correction. First, we reduce the non-uniform background lighting in I_1 and I_2, as it severely contaminates the approximation of the image derivatives $\tilde{I}_s$ using Eq. 12. We adopt the same practice as prior works [8, 20] to filter out the background lighting after measuring the images:

$$I_i^{\text{bck}} = I_i - \frac{1}{K^2} B * I_i, \qquad i = 1, 2, \tag{16}$$

where B is a K×K box filter. We set $K = 21$ in this work based on our experience.

Noise attenuation. The photosensor we use has a significant camera noise. Thus, we apply a Gaussian filter to suppress the noise in the measurements after the aberration correction:

$$I_i^{\text{clean}} = G * I_i^{\text{bck}}, \qquad i = 1, 2. \tag{17}$$

In our experiment, we set the standard deviation of the Gaussian filter G to be 11 pixels.

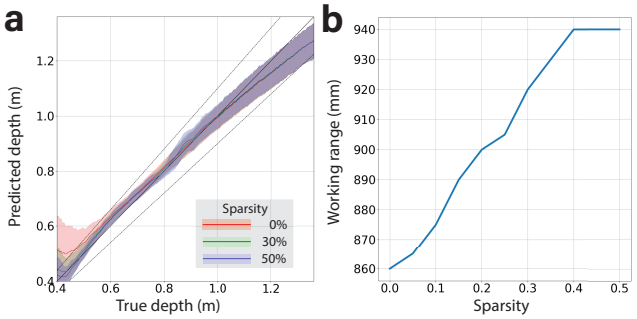

Figure 5. Quantitative analysis of the Focal Split prototype using real captured data. (a) Depth estimation accuracy at different confidence levels. The sparsity indicates the percentage of discarded, least-confident pixels. (b) Working range, defined as the depths where the MAE is smaller than 5% of the true depth, as a function of confidence levels.

Alignment. We determine the homography between the two photosensors from the corresponding SIFT key points from the two images. Then, we compute the approximated image derivatives, $\nabla^2 \tilde{I}$ and $\tilde{I}_s$, via Eq. 12.

Depth estimation. Although our depth estimation algorithm (Eq. 11) can be performed per pixel, we aggregate the image derivatives $\tilde{I}_s$ and $\nabla^2 \tilde{I}$ within an image patch when computing the depth value to attenuate the noise:

$$Z(\boldsymbol{x}) = \frac{a\nabla^2 \tilde{I}(\boldsymbol{x}; s) \left(b\nabla^2 \tilde{I}(\boldsymbol{x}; s) + \tilde{I}_s(\boldsymbol{x}; s) \right) * W}{\left(b\nabla^2 \tilde{I}(\boldsymbol{x}; s) + \tilde{I}_s(\boldsymbol{x}; s) \right)^2 * W}, \tag{18}$$

where the L×L box filter W. We set $L = 21$ in our implementation.

Parameter calibration. The only parameters that need calibration in the entire implementation are the a and b in Eq. 18. We utilize a data-driven approach to determining their values. By moving a front-parallel texture to a series of distances $\{Z_j^*, j = 1, \cdots, J\}$ and capturing an image pair for each distance $\{I_{i,j}, i = 1, 2, j = 1, \cdots, J\}$, and optimizing the following objective function, we obtain the calibrated parameters a and b:

$$\arg\min_{a,b} \sum_{j=1}^{J} \| Z_j^* - Z(\boldsymbol{x}; I_{1,j}, I_{2,j}, a, b) \|^2. \tag{19}$$

5. Real-World Results

5.1. Quantitative Analysis

Working range and depth accuracy. We analyze the working range and depth estimation accuracy using front-parallel textured planes placed at a series of known distances. Fig. 5a-b visualizes the depth estimation accuracy and working range of the Focal Split prototype under different confidence thresholds. The confidence thresholds are

Table 1. System level comparison of monocular passive depth imaging techniques. Only ours achieved untethered depth estimation.

Name	Technique	# Sequential Capture	Depth Map Resolution	Dense Depth Map?	Untethered?	Real Time?	Platform / Power
Newcombe [23]	SfM	>10	640×480	✔	✗	✔	NVIDIA GTX 480 GPU i7 quad-core CPU
Schonberger [29]	SfM	∼10,000	Variable	✗	✗	✗	2.7 GHz Processor 256GB RAM
Tang et al. [30]	DfD	2	5184×3456 or 2464×3280	Initial: ✗ Refine: ✔	✗ ✗	✔ ✗	Two 8-Core 2.6 GHz Xeon CPU 128 GB RAM
Focal Flow [3]	DfDD	3	960×600	✗	✗	✔	2.93 GHz Xeon X5570 CPU
Haim et al. [9]	DfD	1	1920×1080	✔	✗	✗	Nvidia Titan X Pascal GPU
Ikoma et al. [11]	DfD	1	384×384	✔	✗	✔	Unspecified Platform 124 kFLOPs/pixel
Focal Track [7]	DfDD	2	480×300	✗	✗	✔	NVIDIA GeForce GTX 1080 Notebook Graphics Card Intel Core i7 6820HK CPU
COD [20]	DfDD	4	480×300	✗	✗	✗	Intel Core i9-11900K Processor
Ours	DfDD	1	480×360	✗	✔	✔	Raspberry Pi 5 with 2.4 GHz ARM Cortex-A76 Processor 500 FLOPs/pixel, 4.9 W

determined by the overall sparsity of the remaining pixels. By setting the sparsity to 40%, i.e., discarding the 40% least confident pixels, the working range increases from 860 mm to 940 mm, demonstrating the effectiveness of the simple confidence metric.

System characteristics. Table 1 analyzes the specifications of different passive monocular depth estimation systems. Due to the diverse camera setups and computing platforms of these methods, it is challenging to directly and fairly compare these methods in terms of power consumption. Instead, we list the number of sequential captures, depth map resolutions, frame rates, and computing platforms of each method. Compared to methods that generate dense depth maps on more power-hungry platforms, Focal Split provides a complementary option to produce sparse depth maps with low power consumption.

Our Focal Split prototypes generate a 480×360 depth map every 0.47 sec at a power consumption of 4.9 W by running an un-optimized Python script, within which 0.10 sec were used for image measurement and the remaining 0.37 sec were for data processing. Considering the low FLOPs per pixel, the frame rate can be significantly improved if the script is pre-compiled or multi-threading is used. Furthermore, the power consumption of an idling Raspberry Pi is 2.5 W.

Comparison with other DfD algorithms. Focal Split's critical advantage compared to previous computationally efficient DfD algorithms is its snapshot capability. We quantitatively analyze this advantage in Tab. 2. We adopt the front parallel textured planes as the scene and use the Focal Split prototype to resemble several other DfD systems, i.e., Focal Flow [3], Tang et al. [30], and Focal Track [7]. Focal Flow requires three input images with relative motion in between. Thus, we use our prototype to capture three I_1 consecutively while the target is translated axially by 5 mm between the adjacent measurements. Focal Track uses two sequentially measured images of a static scene with different focal planes. Thus, we measure I_1 and I_2 at different time stamps and align them using precalibrated homography as the input data. We deliberately move the target when capturing I_1 and I_2 to resemble a dynamic scene. We use the same input data as Focal Track for Tang et al. Table 2 clearly shows the degradation of depth estimation accuracy of these methods when the scene is dynamic, while Focal Split's accuracy remains invariant under noise.

Table 2. Quantitative comparison between the proposed method and other computationally efficient DfD algorithms using real data. We use the Focal Split prototype to capture the required data for each method. When the scene is static, Focal Track's algorithm is effectively equivalent to ours. Thus, both methods have the same accuracy and working range. However, when the target is dynamic, Focal Track and Tang et al. significantly degrade, while ours remains constant thanks to its snapshot functionality. Focal Flow is designed for dynamic scenes, but its accuracy and working range are both worse than ours. See details about the data collection in Sec. 5.2.

Method	Static Scenes		Dynamic Scenes	
	MAE (mm)	Working Range (m)	MAE (mm)	Working Range (m)
Focal Flow [3]	-	-	179.25	0.600
Tang et al. [30]	109.97	0.355	316.78	0.145
Focal Track [7]	41.82	0.860	107.69	0.295
Ours	**41.82**	**0.860**	**41.82**	**0.860**

5.2. Depth Maps

Fig. 6 shows sample depth maps generated by the Focal Split prototype. The confidence effectively filters out unreliable depth predictions. Besides traditional textured objects, our method can utilize any intensity changes in the images, not necessarily textures, as cues to measure depth.

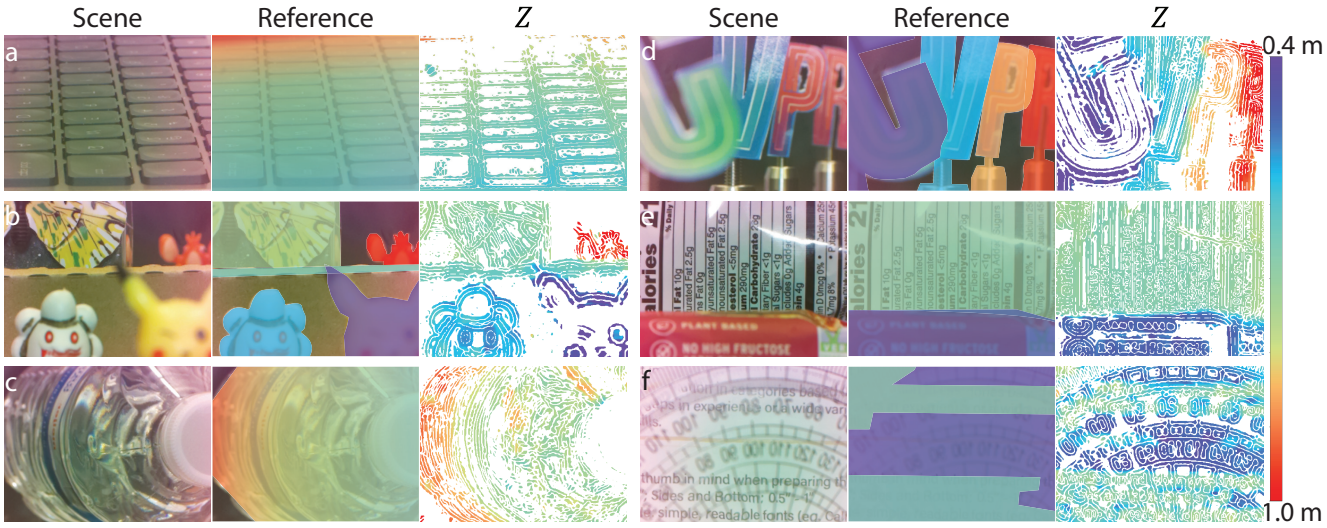

Figure 6. Sample depth maps captured by Focal Split. The depth maps are filtered by the confidence metric with a constant confidence threshold C_{thre}. The reference depth maps are manually measured to provide a qualitative evaluation. More results are in the supplementary.

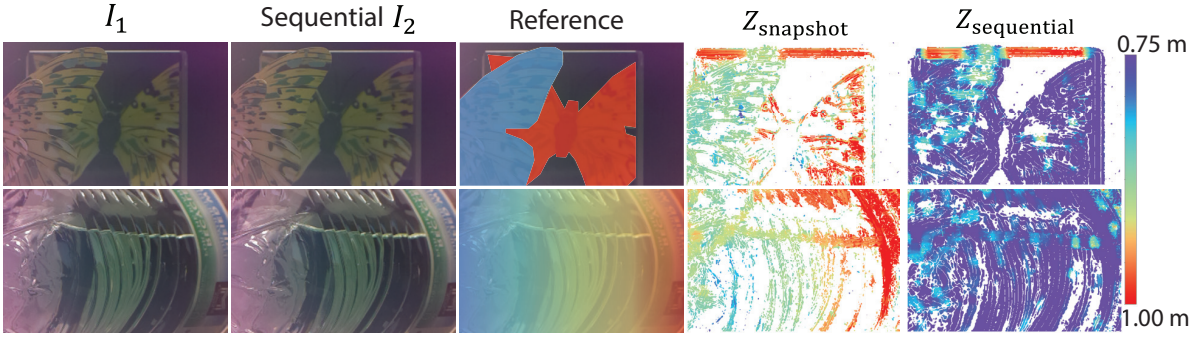

Figure 7. Depth estimation of dynamic scenes using snapshot vs. sequentially measured I_1 and I_2. The symbols Z_{snapshot} and $Z_{\text{sequential}}$ represent the depth maps generated from each capture method, respectively. The depth maps are visualized with the same confidence threshold. The depth map is completely contaminated in $Z_{\text{sequential}}$, caused by the displacement of objects in the two frames due to sequential measurement. Furthermore, these artifacts cannot be identified and removed by the confidence metrics. This experiment demonstrates the critical advantage of snapshot measurement for depth from differential defocus.

Fig. 6c demonstrates using caustics in the water bottle to estimate the depth values, which could be challenging for stereo-based solutions as the caustics are view-dependent. Fig. 6f demonstrates an interesting scenario where the foreground, a protractor, is semi-transparent. In this scenario, traditional dense depth maps are insufficient to represent the scene structure. At each pixel, Focal Split outputs the depth values of the surface with stronger textures in the area centered around it, allowing simultaneous depth estimation for both foreground and background. Meanwhile, a current limitation of Focal Split is that depth estimation becomes inaccurate when both foreground and background have strong textures. Future work includes the prediction of such regions and independently predicting the foreground and background depth values. More depth maps can be found on the project page listed in the abstract.

Besides, we also compare the depth map quality when the input images, I_1 and I_2, are captured in a snapshot vs. sequentially when the target is displaced. As shown in Fig. 7, the sequential measurement clearly results in contaminated depth map measurements compared to snapshot measurements.

Acknowledgement. This research was partially supported by the National Science Foundation under awards numbers CNS-2145584, CNS-2400463, and CIF-2431505. We would also like to acknowledge support by the Alfred P. Sloan Foundation, VMWare, and Catherine M. and James E. Allchin. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or other supporters.

References

[1] Emma Alexander. *A theory of depth from differential defocus*. PhD thesis, Harvard University, 2019. 1, 2, 4, 5

[2] Emma Alexander, Qi Guo, Sanjeev Koppal, Steven Gortler, and Todd Zickler. Focal flow: Measuring distance and velocity with defocus and differential motion. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 667–682. Springer, 2016. 1, 3

[3] Emma Alexander, Qi Guo, Sanjeev Koppal, Steven J Gortler, and Todd Zickler. Focal flow: Velocity and depth from differential defocus through motion. *International Journal of Computer Vision*, 126:1062–1083, 2018. 1, 2, 3, 7

[4] Alexander W Bergman, David B Lindell, and Gordon Wetzstein. Deep adaptive lidar: End-to-end optimization of sampling and depth completion at low sampling rates. In *2020 IEEE international conference on computational photography (ICCP)*, pages 1–11. IEEE, 2020. 2

[5] Xingshuai Dong, Matthew A Garratt, Sreenatha G Anavatti, and Hussein A Abbass. Towards real-time monocular depth estimation for robotics: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):16940–16961, 2022. 2

[6] Jason Geng. Structured-light 3d surface imaging: a tutorial. *Advances in optics and photonics*, 3(2):128–160, 2011. 2

[7] Qi Guo, Emma Alexander, and Todd Zickler. Focal track: Depth and accommodation with oscillating lens deformation. In *Proceedings of the IEEE international conference on computer vision*, pages 966–974, 2017. 1, 2, 3, 7

[8] Qi Guo, Zhujun Shi, Yao-Wei Huang, Emma Alexander, Cheng-Wei Qiu, Federico Capasso, and Todd Zickler. Compact single-shot metalens depth sensors inspired by eyes of jumping spiders. *Proceedings of the National Academy of Sciences*, 116(46):22959–22965, 2019. 1, 3, 6

[9] Harel Haim, Shay Elmalem, Raja Giryes, Alex M Bronstein, and Emanuel Marom. Depth estimation from a single image using deep learned phase coded mask. *IEEE Transactions on Computational Imaging*, 4(3):298–310, 2018. 3, 7

[10] Rostam Affendi Hamzah and Haidi Ibrahim. Literature survey on stereo vision disparity map algorithms. *Journal of Sensors*, 2016(1):8742920, 2016. 3

[11] Hayato Ikoma, Cindy M Nguyen, Christopher A Metzler, Yifan Peng, and Gordon Wetzstein. Depth from defocus with learned optics for imaging and occlusion-aware depth estimation. In *2021 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2021. 3, 7

[12] *Intel RealSense Product Family D400 Series*. Intel, 2020. Rev. 9. 2

[13] San Jiang, Cheng Jiang, and Wanshou Jiang. Efficient structure from motion for large-scale uav images: A review and a comparison of sfm tools. *ISPRS Journal of Photogrammetry and Remote Sensing*, 167:230–251, 2020. 3

[14] Iman Abaspur Kazerouni, Luke Fitzgerald, Gerard Dooly, and Daniel Toal. A survey of state-of-the-art on visual slam. *Expert Systems with Applications*, 205:117734, 2022. 3

[15] Hamid Laga, Laurent Valentin Jospin, Farid Boussaid, and Mohammed Bennamoun. A survey on deep learning techniques for stereo-based depth estimation. *IEEE transactions on pattern analysis and machine intelligence*, 44(4):1738–1764, 2020. 3

[16] MF Land. Structure of the retinae of the principal eyes of jumping spiders (salticidae: dendryphantinae) in relation to visual optics. *The Journal of Experimental Biology*, 51(2): 443–470, 1969. 1

[17] Nalpantidis Lazaros, Georgios Christou Sirakoulis, and Antonios Gasteratos. Review of stereo vision algorithms: from software to hardware. *International Journal of Optomechatronics*, 2(4):435–462, 2008. 3

[18] Anat Levin, Rob Fergus, Frédo Durand, and William T Freeman. Image and depth from a conventional camera with a coded aperture. *ACM transactions on graphics (TOG)*, 26 (3):70–es, 2007. 3

[19] Zhimin Lu, Jue Wang, Zhiwei Li, Song Chen, and Feng Wu. A resource-efficient pipelined architecture for real-time semi-global stereo matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(2):660–673, 2021. 3

[20] Junjie Luo, Yuxuan Liu, Emma Alexander, and Qi Guo. Depth from coupled optical differentiation. *arXiv preprint arXiv:2409.10725*, 2024. 1, 3, 6, 7

[21] Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. Generic and real-time structure from motion using local bundle adjustment. *Image and Vision Computing*, 27(8):1178–1193, 2009. 3

[22] Takashi Nagata, Mitsumasa Koyanagi, Hisao Tsukamoto, Shinjiro Saeki, Kunio Isono, Yoshinori Shichida, Fumio Tokunaga, Michiyo Kinoshita, Kentaro Arikawa, and Akihisa Terakita. Depth perception from image defocus in a jumping spider. *Science*, 335(6067):469–471, 2012. 1

[23] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011. 7

[24] Onur Özyeşil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion*. *Acta Numerica*, 26:305–364, 2017. 3

[25] Francesco Pittaluga, Zaid Tasneem, Justin Folden, Brevin Tilmon, Ayan Chakrabarti, and Sanjeev J Koppal. Towards a mems-based adaptive lidar. In *2020 International Conference on 3D Vision (3DV)*, pages 1216–1226. IEEE, 2020. 2

[26] Luca Puglia, Mario Vigliar, and Giancarlo Raiconi. Real-time low-power fpga architecture for stereo vision. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 64 (11):1307–1311, 2017. 3

[27] Thinal Raj, Fazida Hanim Hashim, Aqilah Baseri Huddin, Mohd Faisal Ibrahim, and Aini Hussain. A survey on lidar scanning mechanisms. *Electronics*, 9(5):741, 2020. 2

[28] Yoav Y Schechner and Nahum Kiryati. Depth from defocus vs. stereo: How different really are they? *International Journal of Computer Vision*, 39:141–162, 2000. 2

[29] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 3, 7

[30] Huixuan Tang, Scott Cohen, Brian Price, Stephen Schiller, and Kiriakos N Kutulakos. Depth from defocus in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2740–2748, 2017. 7

[31] Zaid Tasneem, Charuvahan Adhivarahan, Dingkang Wang, Huikai Xie, Karthik Dantu, and Sanjeev J Koppal. Adaptive fovea for scanning depth sensors. *The International Journal of Robotics Research*, 39(7):837–855, 2020. 2

[32] Zaid Tasneem, Giovanni Milione, Yi-Hsuan Tsai, Xiang Yu, Ashok Veeraraghavan, Manmohan Chandraker, and Francesco Pittaluga. Learning phase mask for privacy-preserving passive depth estimation. In *European Conference on Computer Vision*, pages 504–521. Springer, 2022. 3

[33] Beau Tippetts, Dah Jye Lee, Kirt Lillywhite, and James Archibald. Review of stereo vision algorithms and their suitability for resource-limited systems. *Journal of Real-Time Image Processing*, 11:5–25, 2016. 3

[34] Changyin Zhou, Stephen Lin, and Shree K Nayar. Coded aperture pairs for depth from defocus and defocus deblurring. *International journal of computer vision*, 93:53–72, 2011. 2, 3