Passive Micron-scale Time-of-Flight with Sunlight Interferometry: Supplementary Material

Alankar Kotwal[1], Anat Levin[2], and Ioannis Gkioulekas[1]
[1]Carnegie Mellon University, [2]Technion

1. Implementation details

We discuss implementation details of our sunlight interferometry setup (replicated in Figure 1), and provide a full parts list in Table 1. We take inspiration from the interferometer designs by Gkioulekas et al. [1] and Kotwal et al. [2], and modify them to use sunlight and achieve robustness to environmental conditions for outdoor operation. For alignment of optical components, we use the methods described by Gkioulekas et al. [1] and Kotwal et al. [2].

Vibration isolation. The factor most detrimental to the quality of our results is vibrations in uncontrolled outdoor conditions. Vibrations arise from multiple sources: (a) mechanical vibrations from the ground that propagate up the cart because it has no suspension; (b) movement of the optical components on the table with respect to each other; and (c) lateral and axial movement of the scene and reference and tracking mirrors induced by strong winds.

To mitigate (a), we place our setup on a passively damped honeycomb optical breadboard from Thorlabs on the upper level of our utility cart. We found in experiments that using the breadboard and eliminating any other sources of vibrations (such as the computer in Figure 1(b) and cables connected to the various components) on the upper level of the cart is sufficient. We tightly route the cables so they do not move parts of the setup. With the damped breadboard, vibrations from the lower level of the cart do not travel to the optical setup, so any noisy equipment such as the reference arm translation stage controller (the white cabinet in Figure 1(b)) can be placed on the lower level.

To mitigate (b), we modified the interferometer design from Kotwal et al. [2] so as to tightly clamp down the optical components to each other and to the breadboard. We achieved this by connecting the beamsplitters and the cameras with a $30\,\mathrm{mm}$ cage system using components from Thorlabs. It is not practical to connect the tracking and reference mirrors similarly, so we clamped them tightly to the breadboard using mounting components from Thorlabs.

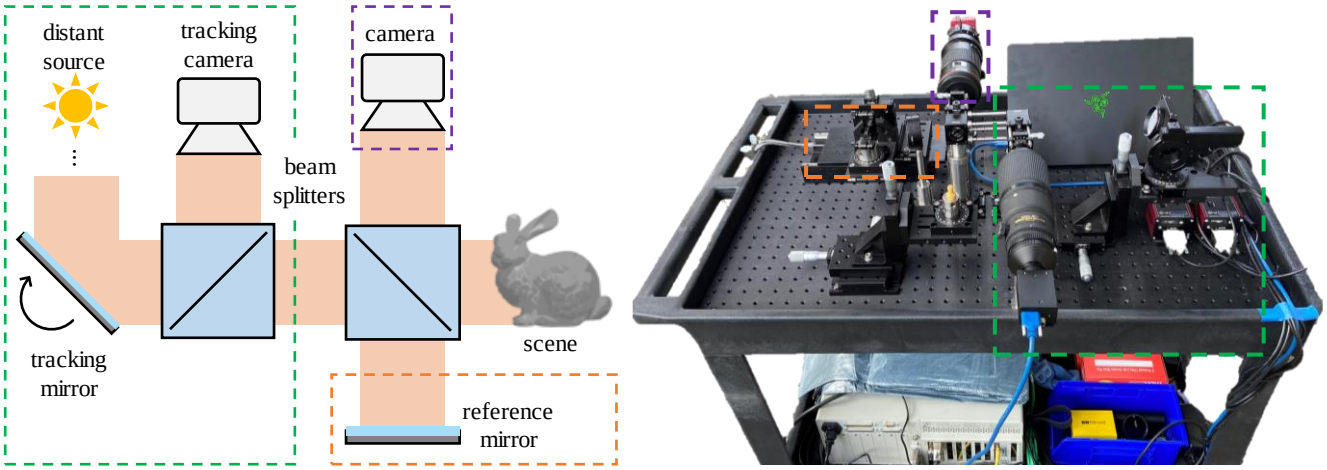
We can mitigate mirror movement in (c) by choosing high-resistance translation stages and high-torque rotation stages. We discuss the properties of our stages below. We found that our setup can tolerate winds as fast as 25 mph.

We estimated the amplitude of vibrations on the setup surface by observing the movement of fringe patterns when we mount mirrors on both scene and reference arms. When we built our setup on an undamped optical breadboard on the cart, we measured vibration amplitude to be around $10\,\mu\mathrm{m}$. By contrast, when we used a passively-damped breadboard, vibration amplitude was below a micrometer (though still orders of magnitude larger than what we measured indoors on an optical table with pneumatic isolation). For the coherence lengths and exposure times we use, we found empirically that this was sufficiently small to allow high-quality depth sensing.

Tracking the Sun. As we mention in the main paper, it is important to track the Sun and center the sunlight beam along the optical axis of the system several times during acquisition. To keep the distribution of sunlight across the field of view uniform, we found that it is necessary to recenter the beam every 100 intensity images. In the duration of capturing 100 images at $50\,\mathrm{ms}$ per image, the Sun moves $0.02\,^\circ$ in the sky. We perform the tracking using two motorized precision rotation stages from Thorlabs, one for the azimuth and one for the altitude of the mirror. These stages have minimum incremental motions of $0.03\,^\circ$, making them perfect for our setup. We found in tests outdoors that the torque provided by these rotation stages was enough to keep the tracking mirror stable in winds as fast as $25\,\mathrm{mph}$. As the control system, we found that a proportional controller with k_p set as half the sensitivity of pixel displacements to rotation stage angles was sufficient.

Reference translation stage. To ensure that we place the reference mirror at the desired positions accurately enough for micron-scale resolutions, we need a translation stage that has a minimum incremental translation less than one micron. We use the XMS160 translation stage from Newport that has a minimal incremental translation of $10\,\mathrm{nm}$. In addition, this stage guarantees low-noise and high-resistance operation, preventing loss of interference contrast due to reference mirror position noise caused by mechanical vibrations and flowing wind.

Beamsplitters. We use thin plate beamsplitters from Thorlabs. The beamsplitter sending light to the tracking camera is a 10:90 (R:T) beamsplitter, reflecting 10% of the input light to the tracking camera. The imaging beamsplitter is a 50:50 (R:T) beamsplitter. We choose plate beamsplitters over cube beamsplitters because cubes cause significant interreflections, and over pellicle beamsplitters because of



(a) schematic (b) physical prototype

Figure 1. **(a)** Schematic and **(b)** physical prototype of the sunlight interferometry setup on a utility cart.

their tendency to distort due to airflow.

Mirrors. We use high-quality protected aluminum mirrors of guaranteed $\lambda/4$ flatness to ensure a uniform phase distribution throughout the sunlight beam.

Spectral filter. As mentioned in the main paper, we control the spectral bandwidth of sunlight using a spectral filter. We choose the spectral filter to balance light efficiency, acquisition time, temporal coherence length, and signal to noise ratio. If we increase the spectral filter bandwidth, the acquisition time decreases, the signal to ratio increases, and the temporal coherence length decreases. The latter makes depth recovery more sensitive to vibrations from the environment. We found in experiments that a spectral filter with central wavelength $550\,\mathrm{nm}$ and bandwidth $20\,\mathrm{nm}$ works best with out setup.

Alignment and calibration. In addition to handling the factors detrimental to interferometric depth reconstruction that we detailed above, we need to deal with the imperfections within the system's construction. The stability that the cage system provides comes at the loss of the capability to finely adjust the orientations and positions of individual components. We need to calibrate for the errors caused by misalignment. In addition, the tracking and control system parameters are critical to set properly for Sun tracking to work fast enough without becoming unstable. With this in view, there are eleven important calibration parameters in the setup for accurate light delivery and tracking stability.

1. Tracking camera principal point: Due to small misalignments in the cage system, when the tracking mirror deflects light perfectly parallel to the optical axis of the system, the image of the Sun at infinity does not appear at the center of the tracking damera sensor. We manually adjust the tracking mirrors to make sure that the shadows of the cage system rods on the beamsplitter mounts disappear. The position of center of the Sun image at this tracking mirror position is the principal point of the tracking camera.
2. Tracking motor speeds and accelerations: The tracking mirror rotation stages have controllable limits on angular speeds and accelerations. We tune these to achieve the minimum incremental motion at the principal point as fast as possible while still being stable.
3. Sensitivity of pixel displacements on the camera to tracking mirror angular displacements: To get a ballpark number for the proportional controller gain, we estimate this sensitivity by rotating the tracking mirror in both dimensions by the minimum incremental rotation of the tracking mirror stages and measure the displacement of the Sun image center.
4. Proportional controller gains: Finally, the proportional controllers we use to output control signals to the rotation stages need to be as fast as possible without overshooting. Overshooting can be catastrophic, because a large overshoot leads the Sun out of the field of view of the tracking camera.
5. Time between tracking adjustments: The movement of the Sun causes spatial intensity non-uniformities in the captured scan. To minimize this while considering the minimum incremental motion of the rotation stages, we calibrate how many scan images to capture between adjustments using the exposure time per image.

Temperature conditions. We conducted experiments over a wide range of temperatures: 3 °C – 20 °C. We found that, at least in this temperature range, we did not need to recalibrate to account for temperature changes. Off-the-self optics and optomechanical components are generally designed for this temperature range, and therefore their properties remain stable as temperature changes.

Table 1. List of major components used in the optical setup in Figure 1(b).

description	quantity	model name	company
3' × 2' utility cart	1	https://www.amazon.com/dp/B001602VI2	Rubbermaid
passively damped optical breadboard	1	B2436FX	Thorlabs
2.56" motorized precision rotation stage	1	PRMTZ8	Thorlabs
1" motorized precision rotation stage	1	PRM1Z8	Thorlabs
K-Cube brushed DC servo motor controller	2	KDC101	Thorlabs
2" round protected aluminum mirror	1	ME2-G01	Thorlabs
25 × 36 mm plate beamsplitter, 10:90 (R:T)	1	BSN10R	Thorlabs
25 × 36 mm plate beamsplitter, 50:50 (R:T)	1	BSW10R	Thorlabs
300 mm compound lens	2	AF Micro Nikkor 300mm 1:4 D IF-ED	Nikon
tracking camera	1	Grasshopper3 USB3 GS3-U3-41C6M-C	FLIR
imaging camera	1	Blackfly S USB3 BFS-U3-122S6M-C	FLIR
1" round protected aluminum mirror	1	ME1-G01	Thorlabs
2" absorptive neutral density filter kit	1	NEK03	Thorlabs
ultra-precision linear motor stage, 16 cm travel	1	XMS160	Newport Corporation
ethernet driver for linear stage	1	XPS-Q2	Newport Corporation
550 ± 20 nm bandpass spectral filter	1	FB550-40	Thorlabs

2. Visualizations of direct light-in-flight

As Equation (9) from the main paper shows, the function $\tau(x, l_m)$ we estimate from our depth recovery pipeline is the direct-only transient response of the scene to incoming illumination. We can then visualize slices of the propagation of direct-only light in the scene as a stack of images of τ. We show some of these slices in Figure 2 for the scenes in Figure 7 of the main paper.

3. Data and reconstruction code

To facilitate reproducibility, we provide in the project website all of the data and scripts used to generate results in the main paper. In addition, Figure 3 provides Matlab code for acquiring measurements with the optical setup of Figure 1(b). Figure 4 provides code for tracking the Sun during acquisition, and Figure 5 provides code for recovering direct-only transients as explained in the post-processing section of the main paper.

References

[1] Ioannis Gkioulekas, Anat Levin, Frédo Durand, and Todd Zickler. Micron-scale light transport decomposition using interferometry. *ACM TOG*, 2015. 1

[2] Alankar Kotwal, Anat Levin, and Ioannis Gkioulekas. Interferometric transmission probing with coded mutual intensity. *ACM TOG*, 2020. 1

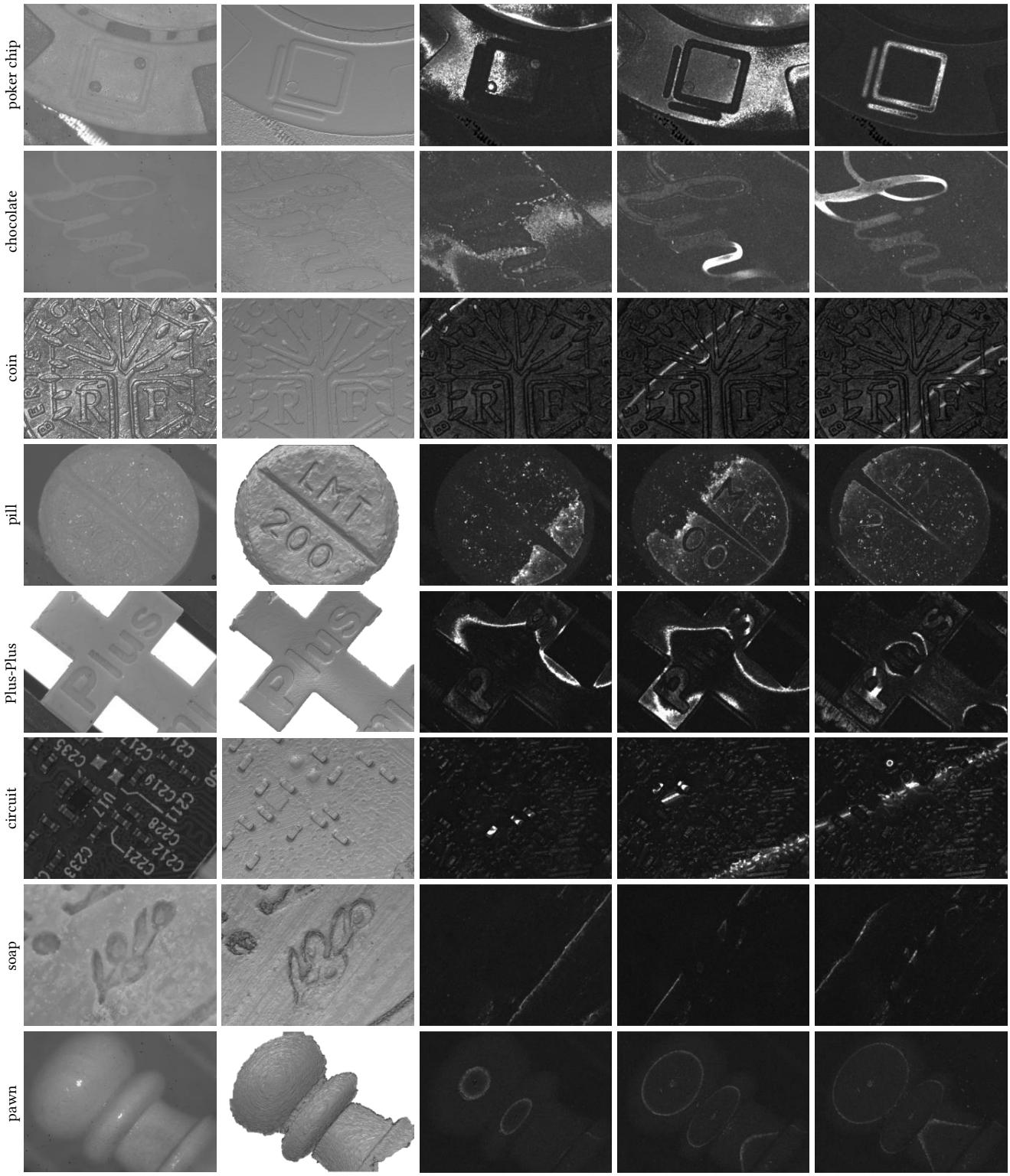


Figure 2. **(b)** Depth reconstructions using sunlight, rendered as 3D surfaces and **(c)** sample raw frames from the depth scans.

```
function frames = acquire(positions, imageSize, vid, refMotor, ...
                          trackingVid, trackingMotor, adjustEvery, ...
                          trackingSensitivity, trackingKp, trackingTol)

    % Acquire passive interferometry data

    % Reset motor
    refMotor.goto(positions(1));
    resolution = positions(2)-positions(1);

    % Initialize array for frames
    frames = zeros([imageSize numel(positions)], 'uint16');

    % At every position
    for i = 1:numel(positions)

        % Center the sun at every adjustEvery positions
        if mod(i, adjustEvery) == 0
            track(trackingVid, trackingMotor, trackingSensitivity, trackingKp, trackingTol);
        end

        % Record an image at this position
        frames(:, :, i) = getsnapshot(vid);

        % Translate reference arm to the next position and give it time to settle
        refMotor.translate(resolution);
        pause(0.1);

    end

end
```

Figure 3. Matlab code for recovering depth from our measurements

```
function track(vid, motor, sensitivity, kp, tol)

    % Center the sun on the frame

    % One-dimensional tracking for brevity
    % Easily extended to two dimensions
    err = Inf;
    correction = 0;

    % Loop unless you can tolerate error
    while abs(err) > tol

        % Translate tracking motor by previous correction
        motor.translate(correction);

        % Get sun center coordinates
        image = getsnapshot(vid);
        xx = getSourceCenter(image);

        % Correct based upon the error between the image center and sun center
        err = xx-size(image)/2;
        correction = -kp*sensitivity*err;
    end

end
```

Figure 4. Matlab code for sun tracking

```
function [depth, direct] = process(frames, positions, spatialWindow, temporalWindow)

    % Reconstruct depth from passive interferometry measurements
    % frames: HxWxN array of measurements from the camera
    % positions: set of positions of the reference arm
    % spatialWindow: blur kernel width for blurring speckle
    % temporalWindow: kernel for estimating interference-free frames

    % Convert uint16 frames from camera into double (or float in case of memory overflows)
    frames = im2double(frames);

    halfTemporalWindow = temporalWindow/2;

    % Blur 'temporalWindow' frames along the positions to estimate per-position
    % interference-free images
    interferenceFree = convn(frames, ones(1, 1, temporalWindow)/temporalWindow, 'same');

    % Pad the interference-free images to remove convolution edge artifacts
    interferenceFree(:, :, 1:halfWindow) = ...
                    repmat(interferenceFree(:, :, halfWindow+1), [1 1 halfWindow]);
    interferenceFree(:, :, end-(halfWindow-1):end) = ...
                    repmat(interferenceFree(:, :, end-halfWindow), [1 1 halfWindow]);

    % Subtract interference-free images to get interference-only images
    interference = (frames-interferenceFree).^2;

    % Blur with a kernel 'spatialWindow' in size
    interference = sqrt(imgaussfilt(interference, spatialWindow));

    % Get maximums of interference as ballistic light path image, and positions at maxima
    % as depth
    [direct, idxs] = max(interference, [], 3);
    depth = positions(idxs);

end
```

Figure 5. Matlab code for acquiring measurements