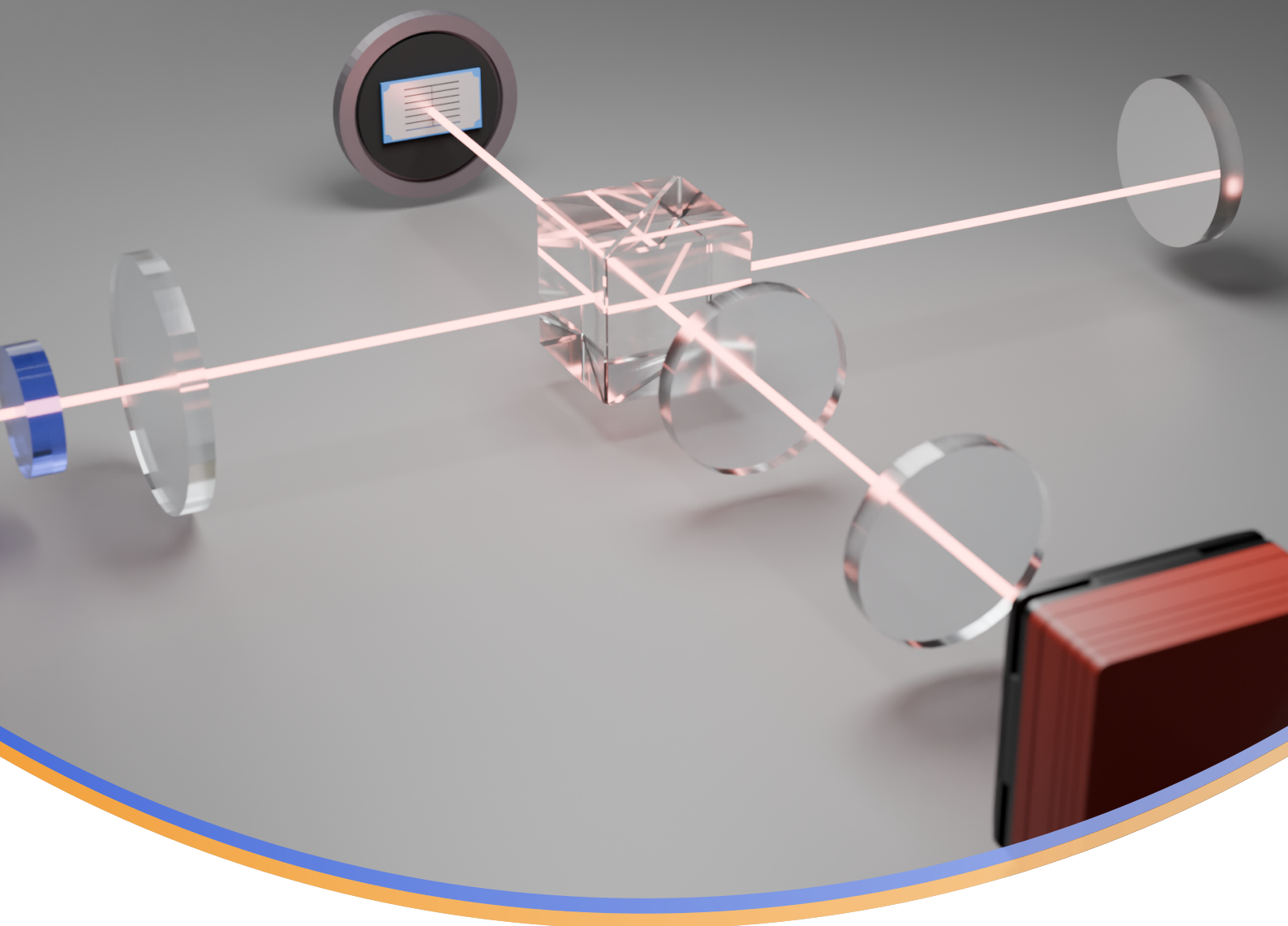




Development of a Full-Field Interferometric System to Detect Low-Frequency Vibrations in Micro-Acoustic Components

Master's thesis in partial fulfilment of the requirements for the degree of Master of Science (M.Sc.)

Qualcomm

conducted at Qualcomm RF360

Entwicklung eines Vollfeld-Interferometriemesssystems zur Erkennung niederfrequenter Schwingungen in Mikroakustik-Komponenten

Masterarbeit zur Erlangung des akademischen Grades Master of Science (M.Sc.)

MASTERARBEIT

Gemäß §6 Studien- und Prüfungsordnung (SPO) vom 15.02.2023 für den Masterstudiengang Mikro- und Nanotechnik an der Hochschule für angewandte Wissenschaften München sowie §26 Allgemeine Studien- und Prüfungsordnung (ASPO) vom 15.02.2023 der Hochschule für angewandte Wissenschaften München in derzeit gültiger Fassung.

VERFASSER/AUTHOR:

KONTAKT/CONTACT:

MATRIKELNR.:

Alexander Taka Edelemann

alexanderedelmann.com

71578413

BETREUER/SUPERVISOR (QUALCOMM)

ERSTPRÜFER/FIRST EXAMINER

ZWEITPRÜFER/SECOND EXAMINER

ABGABEDATUM/DATE OF SUBMISSION

Dr. Kimmo Kokkonen

Prof. Dr. Thomas Hellerer

Prof. Dr. Datong Wu

Apr. 30, 2026

A b s t r a c t

This thesis presents the development of a custom full-field interferometric measurement system that leverages software-driven stabilization for the vibration analysis of micro-electro-mechanical systems (MEMS). It outlines in detail the principles underlying the operation of the system and the processing of the measurement data. Experimental validation on in-house MEMS samples demonstrates successful stabilized measurements and the reconstruction of physically consistent amplitude and phase maps. The system achieves sub-nanometer displacement sensitivity, with a practical noise floor of approximately 200 pm. Moreover, insight into the stabilization code architecture is given, realized in both MATLAB and Python.

Kurzfassung

Diese Arbeit präsentiert die Entwicklung eines maßgeschneiderten Vollfeld-Interferometriemesssystems, das softwaregesteuerte Stabilisierung für die Schwingungsanalyse von mikroelektromechanischen Systemen (MEMS) nutzt. Es werden ausführlich die Prinzipien, die dem Betrieb des Systems und der Verarbeitung der Messdaten zugrunde liegen, beschrieben. Die experimentelle Validierung an hausinternen MEMS-Proben zeigt erfolgreiche stabilisierte Messungen und die Rekonstruktion physikalisch konsistenter Amplituden- und Phasenkarten. Das System erreicht eine Verschiebungsempfindlichkeit im Subnanometerbereich, mit einer praktischen Rauschuntergrenze von ungefähr 200 pm. Darüber hinaus wird Einblick in die Architektur des Stabilisierungscode gegeben, der sowohl in MATLAB als auch in Python realisiert ist.

Contents

1	Introduction	1
1.1	Motivation and Problem Context	1
1.2	Objectives and Scope	3
1.3	Thesis Structure	3
2	Background and Theory	5
2.1	Fundamentals of Optical Interferometry	5
2.1.1	Superposition of Waves	5
2.1.2	Interferometry	6
2.1.3	Coherence of Light Sources	8
2.2	Vibration measurements and signal processing	9
2.2.1	Stroboscopic Detection of Vibrations	9
2.2.2	Discrete Fourier Transform	9
2.2.3	Shot Noise	11
2.3	Summary and Further Reading	13
3	Measurement System Development	15
3.1	System Architecture	15
3.1.1	System Components	15
3.1.2	Stabilization	16
3.1.3	Operating Principle	17
3.2	Measurement Procedure	18
3.2.1	Quadrature Point Detection (Calibration)	18
3.2.2	Measurement Principle and Parameters	19
3.2.3	Error Sources and Mitigation	21
3.3	Summary	21
4	Experimental Validation and Results	25
4.1	MEMS Cantilever Sample	25
4.2	Amplitude Extraction Procedure	25
4.3	Frequency Response Measurement	27
4.4	Resonance Bandwidth and Quality Factor	28
4.5	Validation Against Euler–Bernoulli Beam Theory	28
4.6	Noise Analysis	30
4.7	Summary	31
5	Measurement System Software Development	33
5.1	GUI Design and Workflow	33
5.2	MATLAB Implementation	33
5.2.1	Single-thread execution model and implications	33
5.2.2	High-level system architecture	34
5.2.3	Stabilization-aware acquisition and GUI dispatch	36
5.3	Python Implementation	38

5.3.1	High-level system architecture	38
5.3.2	Multi-threaded stabilization-aware acquisition and GUI dispatch	38
5.4	Comparison and Summary	41
6	Conclusions and Future Work	43
6.1	Summary of Achievements	43
6.2	Limitations and Future Directions	43
6.3	Summary of Contributions and Concluding Remarks	44

List of Figures

2.1	Superposition of two harmonic waves ψ_1 and ψ_2 for different relative phases $\Delta\phi$	6
2.2	Michelson interferometer and detection modalities	7
2.3	Low-coherence interferometry	8
2.4	Discrete Fourier Transform and its complex exponential basis	10
2.5	Poisson statistics and discrete intensity detection	12
3.1	Experimental setup and operating principle	15
3.2	Vizualization of LED coherence envelope	16
3.3	Calibration sweep and operating-point selection from representative measurement data	19
3.4	Data acquisition modalities	20
3.5	Measurement workflow	22
4.1	MEMS cantilever array, micrograph, and sensitivity map	26
4.2	Single-pixel frequency response and Lorentzian peak fits	27
4.3	Full-field vibration amplitude and phase at the first three bending modes	28
4.4	Euler-Bernoulli mode-shape fits along the beam centerline	29
4.5	Profile line (at $y = 70$) for different averaging numbers N of the beam at the third bending mode	30
5.1	Javascript React GUI	34
5.2	MATLAB single thread execution model	35
5.3	Software architecture diagram of the MATLAB code base	36
5.4	Python multi-threaded acquisition (timeline)	39

List of Tables

1.1	Common examples of resonant MEMS devices.	2
4.1	Measured resonance frequencies and ratios vs. Euler–Bernoulli predictions.	29
4.2	Fit parameters of measured centerline profiles to Euler–Bernoulli mode shapes	30
5.1	Comparison of MATLAB and Python implementations	42

1 Introduction

1.1 Motivation and Problem Context

Micro-electro-mechanical systems (MEMS) emerged from the broader trend toward device miniaturization and functional integration. In general, MEMS combine micrometer-scale mechanical structures with electrical functionality on a common substrate, typically using batch-fabrication processes derived from semiconductor manufacturing [1, 2]. While the term originally referred primarily to miniaturized mechanical devices, it is now often used more broadly to include microsystems with sensing, actuation, optical, fluidic, or biochemical functionality [2, 3]. Characteristic dimensions usually range from a few micrometers to the millimeter scale, although the exact size range depends on the application and fabrication technology [1].

Commercial MEMS technology gained strong momentum in the 1990s, driven in particular by integrated silicon accelerometers and gyroscopes enabled by surface micromachining and CMOS-compatible fabrication [4, 5, 3]. Early commercial devices were predominantly silicon-based because they could exploit the mature materials, tooling, and process control of the integrated-circuit industry. As the field matured, MEMS technology expanded to include a broader range of structural and functional materials, including piezoelectric thin films, silicon carbide, polymers, and compound semiconductors, thereby enabling improved performance, lower power consumption, and increasingly complex device functionality [6, 3, 7].

Within this landscape, resonant and acoustic MEMS have become particularly important. Micromachined resonators are attractive because of their small footprint, low power consumption, and compatibility with wafer-scale manufacturing, which make them suitable for timing applications, inertial sensing, pressure sensing, and chemical or biological sensing [7, 8]. In parallel, the rapid development of wireless communication systems and the proliferation of mobile and IoT devices have strongly increased the demand for compact, low-cost, and highly integrated radio-frequency front-end components. In this context, surface acoustic wave (SAW) and bulk acoustic wave (BAW) devices play a central role in RF filtering and frequency selection. For example, modern 5G smartphones may contain on the order of one hundred acoustic-wave filters to handle the growing number of frequency bands and the associated coexistence requirements [9, 10].

Representative examples of resonant MEMS are summarized in Table 1.1.

Accurate dynamic characterization is essential both for evaluating new device concepts and for validating analytical and numerical models before deployment. A central figure of merit of resonant devices is the quality factor,

$$Q = 2\pi \frac{\text{energy stored}}{\text{energy dissipated per cycle}},$$

which quantifies how efficiently a resonator stores mechanical energy. High- Q resonators exhibit narrow resonance linewidths and are particularly attractive for filters, oscillators, and precision sensors

Device	Operating principle	Application
MEMS Oscillators	Silicon beam oscillates at a stable resonant frequency (e.g. 32.768 kHz).	Replacement for quartz crystals in watches, laptops, IoT nodes.
Gyroscopes	Vibrating masses experience Coriolis forces when rotated.	Phone screen rotation, drone stabilization, VR/AR systems.
Resonant Pressure Sensors	Membrane tension changes with pressure, shifting resonance frequency.	Aerospace and industrial altimetry; precision pressure sensing.
BAW/SAW Filters	Piezoelectric excitation generates acoustic waves with frequency-selective response.	RF filtering in smartphones, Wi-Fi/5G front-end modules.
Mass Sensors	Resonant frequency shifts when adsorbed mass increases (e.g., molecule binding).	Chemical and biological detection (“e-nose” systems).

Table 1.1 | Common examples of resonant MEMS devices.

because they can provide improved frequency selectivity, enhanced force or mass sensitivity, and superior frequency stability.

A broad range of experimental techniques is available for MEMS characterization, each involving trade-offs in spatial resolution, temporal bandwidth, dynamic range, invasiveness, and measurement throughput. Atomic force microscopy (AFM) and scanning electron microscopy (SEM) provide very high spatial resolution, but AFM is contact-based and therefore potentially invasive for compliant microstructures, while SEM is generally not suited to routine, quantitative vibration measurements under practical operating conditions. For dynamic testing, optical non-contact methods are therefore particularly attractive.

Among optical techniques, scanning laser vibrometry methods are widely established. Within this spectrum Doppler methods (LDV) and interferometric setups stand out. These point-measurement techniques can reach exceptional sensitivity from near-DC up to the MHz–GHz regime [11, 12, 13, 14].

Complementary to scanning laser vibrometry, digital holographic interferometry and related camera-based approaches (e.g., stroboscopic holography, stroboscopic white-light interferometry) provide instantaneous full-field amplitude and phase maps. Such methods deliver dense spatial sampling, capturing complex mode shapes and time-resolved vibration fields without the repeatability challenges associated with point-by-point scanning. In addition, their optical layouts can often be realized with comparatively modest hardware effort, especially in homodyne and stroboscopic configurations.

¹

However, camera-based full-field methods also have important limitations. Because the available optical power is distributed over many pixels, the signal level per pixel is typically much lower than in single-point systems that use dedicated high-sensitivity photodetectors. As a result, per-pixel sensitivity is often lower and measurements can become limited by shot noise, camera read noise, or quantization effects, especially at short exposure times. Published examples of full-field setups report sensitivities of $\lesssim 30$ pm in stabilized stroboscopic setups [15, 16, 17]. In contrast, commercially available single-point systems can reach sub-picometer sensitivities [18], and for example, Erlend Leirset et al. [14] presented a heterodyne interferometer that can reach a noise floor of ~ 7.1 fm.

In the context of this work, the focus lies on the characterization of acoustic MEMS operating predominantly below approximately 1 MHz. Key requirements are full-field visualization of vibration patterns over broad frequency ranges, sub-nanometer displacement sensitivity, and practical usability in a laboratory environment. These requirements motivate the development of a cost-conscious full-field interferometric measurement system. Although commercial turnkey instruments are available, a custom-built platform offers three important advantages: (i) *flexibility*, for example through

¹By contrast, some scanning interferometers—for example the heterodyne setup in Ref. [14]—require multiple RF sources (e.g., six signal generators) for frequency shifting, modulation, and demodulation.

user-selectable magnification, working distance, and illumination conditions; (ii) *insight*, because direct control of coherence, timing, and stabilization facilitates a deeper understanding of error sources and performance limits; and (iii) *upgradability*, since the system can be extended incrementally, for example by improved synchronization electronics, alternative light sources, or modified microscope optics, as future measurement demands evolve.

1.2 Objectives and Scope

The objective of this work is to develop and validate a full-field interferometric measurement system based on low-temporal-coherence LED illumination and software-driven stabilization. A custom software framework is implemented to control all system components, manage synchronization, and provide a user-friendly interface for automated measurements with real-time feedback.

Experimental validation is carried out using in-house-fabricated MEMS devices. The scope of the system—and of this thesis—is limited to the characterization of small-amplitude, out-of-plane vibrations. Large-amplitude motion, nonlinear dynamic behavior, and multi-degree-of-freedom coupling phenomena are outside the present scope and are not addressed.

1.3 Thesis Structure

The remainder of this thesis is organized as follows.

Chapter 2 provides a comprehensive overview of the theoretical foundations of optical interferometry and its relevance to MEMS characterization. It covers the fundamentals of wave interference, coherence properties of light sources, and the principles underlying full-field interferometric measurements.

Chapter 3 describes the design and implementation of the full-field interferometric setup developed in this work. It introduces the system architecture, operating principles, and data-acquisition strategies. Key aspects of the accompanying control and acquisition software are detailed in Chapter 5.

Chapter 4 presents the experimental validation of the system using in-house-fabricated MEMS samples. This chapter outlines the data-processing pipeline used to extract amplitude and phase maps of vibration modes and demonstrates the system's capability to detect low-frequency vibrations with high spatial resolution and sensitivity.

Finally, Chapter 6 summarizes the main findings of this work and discusses potential extensions, improvements, and future research directions enabled by the results presented here.

2 Background and Theory

2.1 Fundamentals of Optical Interferometry

2.1.1 Superposition of Waves

Optical fields—transverse electromagnetic waves propagating through space—can be conveniently represented using complex phasors. For a linearly polarised, monochromatic wave propagating in the z -direction, the complex field can be written as ^{1 2}

$$\psi(z) = A \exp(i(kz + \phi)), \quad (2.1)$$

where A is the field amplitude, $k = 2\pi/\lambda$ is the wavenumber with wavelength λ , and ϕ denotes the initial phase. The wavelength is related to the optical frequency f and propagation speed c by $\lambda = c/f$. In physical measurements only the real part of the field is observable; however, the complex representation greatly simplifies the mathematical treatment of wave superposition.

When two coherent optical fields ψ_1 and ψ_2 (see Sec. 2.1.3 for definition of coherence) overlap spatially, the resulting field is given by their linear superposition

$$\psi(z) = \psi_1(z) + \psi_2(z). \quad (2.2)$$

Assuming both fields share the same optical frequency, the superposition becomes

$$\psi(z) = A_1 \exp(i(kz + \phi_1)) + A_2 \exp(i(kz + \phi_2)) \quad (2.3)$$

$$= \underbrace{(A_1 e^{i\phi_1} + A_2 e^{i\phi_2})}_{A e^{i\phi}} e^{ikz}. \quad (2.4)$$

The resulting field therefore retains the same spatial periodicity as the individual waves, but its amplitude and phase depend on the relative phase difference (Fig. 2.1)

$$\Delta\phi = \phi_1 - \phi_2. \quad (2.5)$$

Constructive interference (maximum amplitude) occurs when the phase difference is an integer multiple of 2π (Fig. 2.1a),

$$\Delta\phi = 2\pi n, \quad n \in \mathbb{Z},$$

while destructive interference (minimum amplitude) occurs for odd multiples of π (Fig. 2.1c),

$$\Delta\phi = (2n + 1)\pi, \quad n \in \mathbb{Z}.$$

¹Since the electric and magnetic fields are linked by Maxwell's equations, it is in most cases sufficient to consider only one field component (conventionally the electric field).

²Only the time-independent (spatial) part of the field is considered here. The full time-dependent field is recovered by multiplying by $\exp(i\omega t)$.

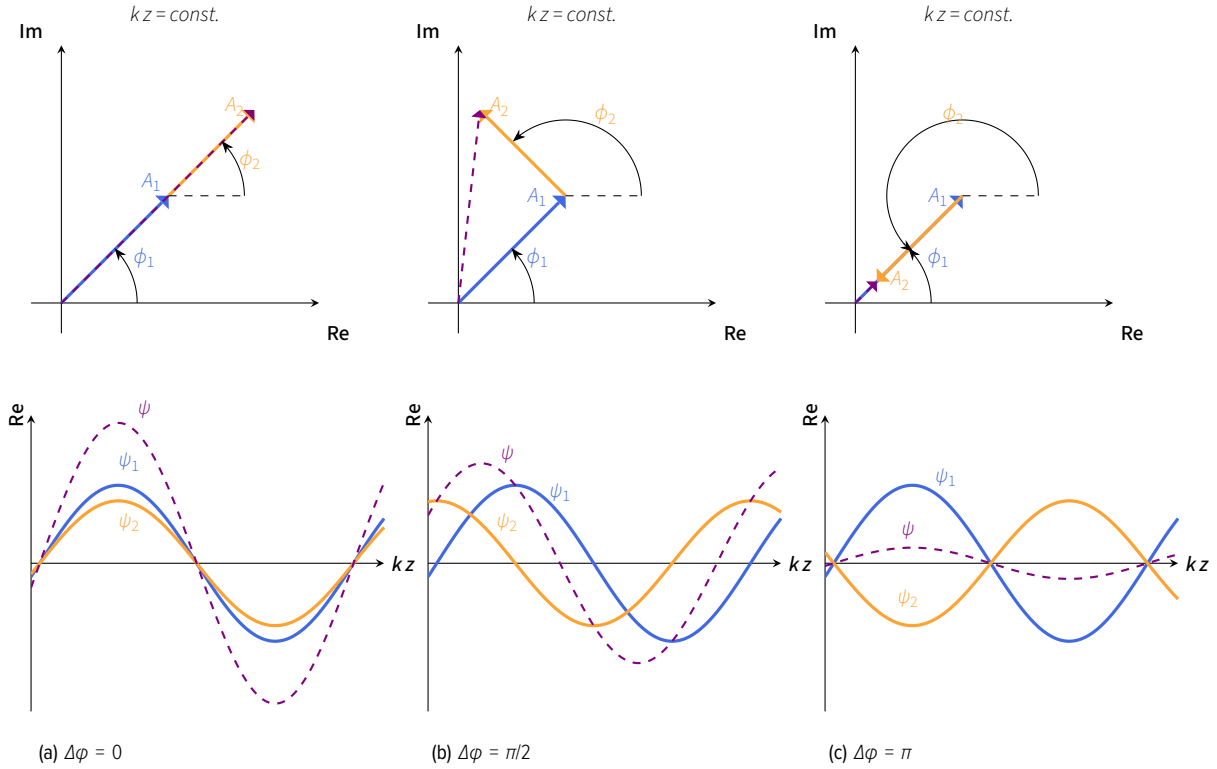


Figure 2.1 | Superposition of two harmonic waves ψ_1 and ψ_2 for different relative phases $\Delta\phi$. For each case, the upper panel shows the phasor representation at fixed $kz = \text{const.}$, where the complex amplitudes of ψ_1 and ψ_2 are added vectorially to yield the resultant field ψ . The lower panel shows the corresponding real parts of ψ_1 , ψ_2 , and ψ as functions of kz . For $\Delta\phi = 0$ (a), the waves interfere constructively and the resultant amplitude is maximal. For $\Delta\phi = \pi/2$ (b), partial constructive interference occurs and the resultant amplitude is reduced. For $\Delta\phi = \pi$ (c), the interference is maximally destructive and the resultant amplitude is minimal; it vanishes only if the two waves have equal amplitudes.

Optical detectors measure intensity which is proportional to the squared magnitude of the total field:

$$I \propto |\psi|^2. \quad (2.6)$$

Substituting the superposed field (Eq. (2.4)) yields ^{3 4}

$$I = |(A_1 e^{i\phi_1} + A_2 e^{i\phi_2})|^2 = |A_1|^2 + |A_2|^2 + 2A_1 A_2 \cos(\Delta\phi). \quad (2.7)$$

Expressing the individual beam intensities as $I_1 = A_1^2$ and $I_2 = A_2^2$ leads to the well-known interference relation

$$I = I_1 + I_2 + 2\sqrt{I_1 I_2} \cos(\Delta\phi). \quad (2.8)$$

2.1.2 Interferometry

Interferometry is an optical measurement technique that exploits the phase-dependent intensity modulation described above (Sec. 2.1.1) to quantify physical parameters such as displacement, distance, surface topography, and refractive-index variations. The fundamental principle involves splitting an incident light beam into two or more paths that traverse different optical path lengths before being recombined⁵. The optical path difference (OPD) between the beams is directly proportional to their

³The squared magnitude of the sum of two complex numbers z_1, z_2 is $|z_1|^2 + |z_2|^2 + 2\text{Re}\{z_1 z_2^*\}$

⁴By symmetry: $\cos(\phi_1 - \phi_2) = \cos(\phi_2 - \phi_1) = \cos(\Delta\phi)$

⁵This work focuses exclusively on homodyne interferometry, in which the interfering beams originate from the same light source and therefore share the same optical frequency.

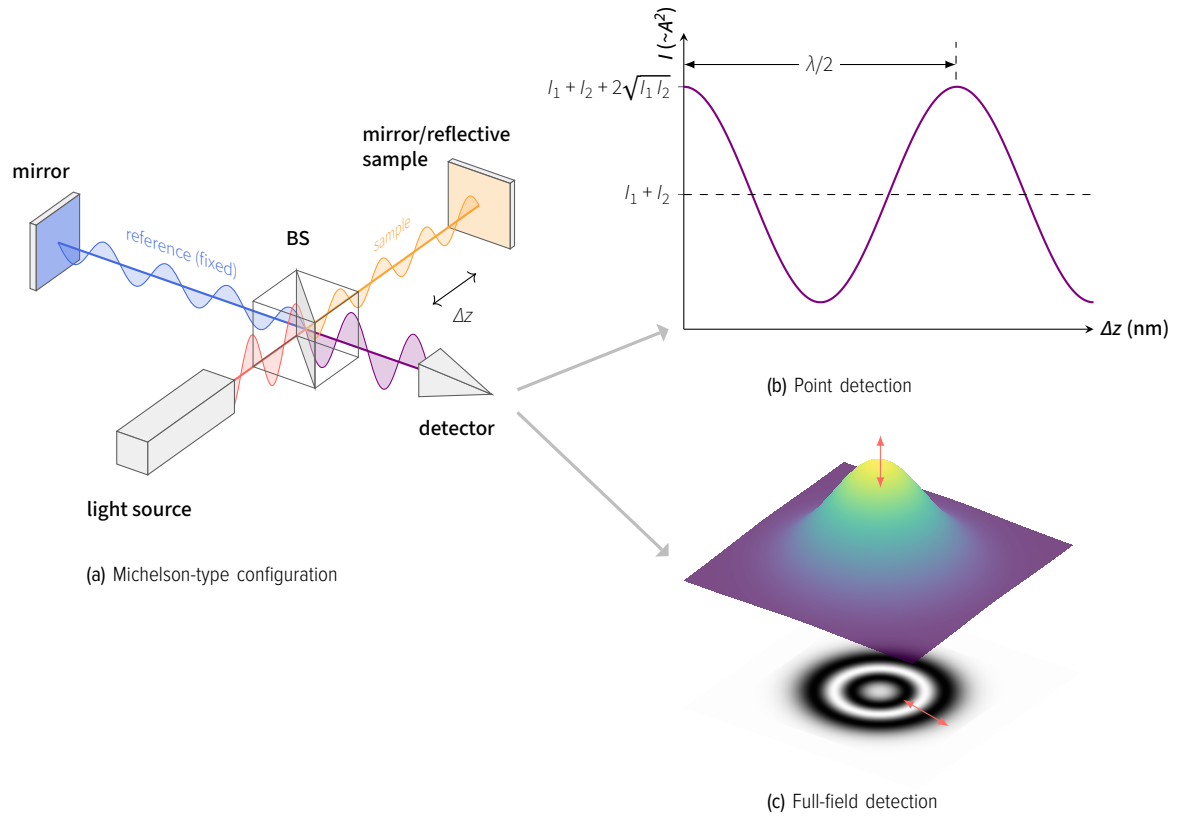


Figure 2.2 | Michelson interferometer and detection modalities. In the Michelson-type interferometer shown in (a), a non-polarizing beam splitter (BS) divides the incident light into a reference arm and a sample arm. The beam in the reference arm is reflected by a mirror with fixed position, while the beam in the sample arm is reflected either by the sample itself or by a second mirror, depending on the measurement task. After returning to the beam splitter, both beams are recombined and directed to the detector. For point detection (b), the detector records the intensity at a single spatial position as a function of Δz , yielding the characteristic sinusoidal interferometric response (Eqs. 2.8 and 2.9). For full-field detection (c), the intensity is recorded simultaneously over an extended area. Spatial variations of Δz then give rise to an interferogram consisting of bright and dark fringes. The illustrated example corresponds to a radially symmetric Δz -distribution, which produces circular interference fringes. Changes in surface height or refractive index modify/shift the fringe pattern.

phase difference $\Delta\phi$ and thus produces measurable intensity variations (Eq. (2.8)), giving interferometric techniques exceptionally high displacement sensitivity.

In full-field interferometry, interference is observed simultaneously over an extended area rather than at a single point, by illuminating the sample with collimated light and recording the resulting pattern with a camera. Spatial variations of the OPD across the field of view give rise to so-called interference fringes. For example, a radially symmetric surface profile produces the alternating circular bright and dark bands shown in Fig 2.2c. In the interferogram, a continuous change in surface height results in fringe movement towards or away from the center. More fringes emerge as larger surface height differences occur, and fewer fringes appear with smaller height differences.

Various interferometric configurations exist—including Michelson, Mach–Zehnder, and Fabry–Pérot interferometers—each offering specific advantages depending on the measurement requirements.

The configuration relevant to this work is the Michelson interferometer, illustrated in Figure 2.2a. A beam splitter divides the incident light into two paths: a reference arm and a sample arm. In the reference arm, the beam is reflected by a fixed mirror, maintaining a constant optical path length. In the sample arm, the optical path length may vary—for example, due to changes in refractive index or propagation distance. Upon recombination at the beam splitter the two beams interfere, and the resulting intensity is recorded by a photodetector.

Because light traverses each arm twice (round trip), an OPD Δz between the sample and reference

arms corresponds to twice the physical displacement. The resulting phase shift is therefore

$$\Delta\varphi = \frac{4\pi}{\lambda} \Delta z. \quad (2.9)$$

2.1.3 Coherence of Light Sources

Coherence theory provides a statistical description of the radiation field produced by a light source in terms of the correlation between field vibrations at different points in space and time.

An ideal monochromatic source emits a perfectly continuous sinusoidal wave. Real sources, however—owing to the discontinuous nature of the emission process—never satisfy this condition; they always radiate over a range of frequencies, although that range can be very narrow. The time interval during which the emitted wave approximates a phase-stable sinusoidal oscillation is a measure of the source's *temporal coherence*. *Spatial coherence*, by contrast, quantifies the phase correlation between different points across a wavefront and is governed by the source size and geometry.

Temporal coherence is characterised by the coherence length l_c , which is inversely related to the spectral bandwidth of the source; for an ideal monochromatic source, l_c would be infinite. For a source with a Gaussian spectral profile, central wavelength λ_0 , and full-width-at-half-maximum (FWHM) spectral bandwidth $\Delta\lambda$, the coherence length is (see Ref. [19] for a derivation)

$$l_c = \frac{2 \ln(2)}{\pi} \frac{\lambda_0^2}{\Delta\lambda} \approx \frac{\lambda_0^2}{2\Delta\lambda}. \quad (2.10)$$

In the context of interferometric measurements, l_c is a fundamental quantity because it determines the maximum optical path difference (OPD) over which interference fringes can be observed. For example, a conventional gas-discharge lamp has a coherence length of a few millimetres, whereas certain stabilised lasers achieve coherence lengths in the tens-of-kilometres range. In general, sources with long coherence lengths also exhibit high spatial coherence.

Although many interferometric techniques employ high coherence illumination such as lasers, two effects can become limiting factors in certain applications:

(i) Slope-sign ambiguity High temporal coherence produces a periodic interference signal, making it impossible to distinguish the direction of a displacement once the OPD exceeds half the wavelength ($\lambda_0/2$).

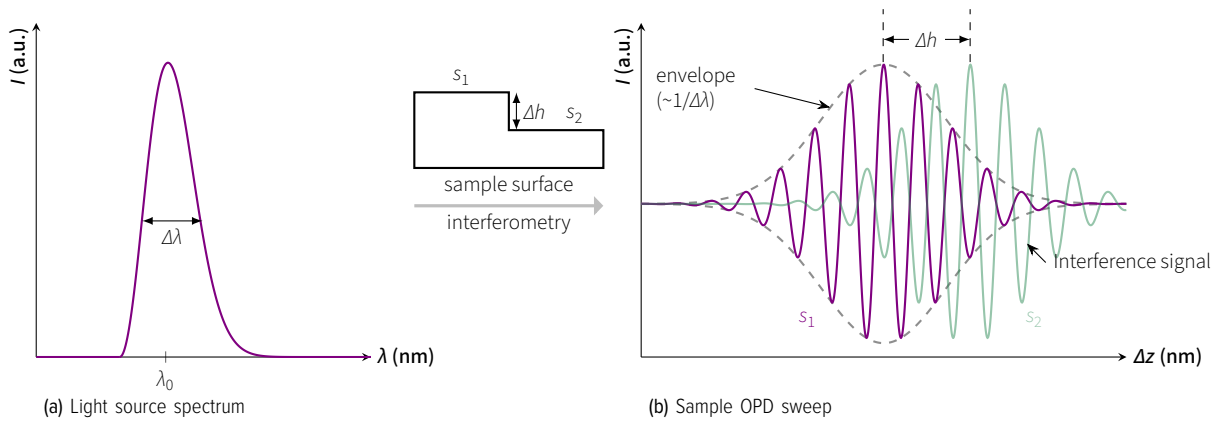


Figure 2.3 | Low-coherence interferometry. Using broad-band light sources (a) for interferometric measurements, the interference curve (Fig. 2.2b) will be enveloped and is only observable in a range $\sim \lambda_0^2/(2\Delta\lambda)$ around the zero OPD. Consequently, if interferometric signals are recorded as a function of Δz for sample surface points at varying heights (b), their height differences are identified as the distance between the curve maxima.

(ii) **Speckle noise** In full-field interferometric configurations, high spatial coherence causes neighbouring rays to interfere with one another, producing unwanted speckle patterns that degrade image quality.

These limitations can be addressed by employing sources with broader spectral bandwidths, such as superluminescent diodes (SLDs) or light-emitting diodes (LEDs). Such sources greatly suppress speckle, thereby enabling high-resolution full-field interferometric imaging. Because of the reduced coherence length, interference is observed only within a narrow region around zero OPD, and the sinusoidal fringe pattern is modulated by an envelope (typically of Gaussian shape) whose peak provides a distinct localisation signature. Interferometric surface profiling, for instance, exploits this property (Fig. 2.3): the envelope maximum marks the position of zero OPD at each point on the surface, enabling the detection of topographical variations without the $\lambda_0/2$ ambiguity inherent in monochromatic techniques.

2.2 Vibration measurements and signal processing

2.2.1 Stroboscopic Detection of Vibrations

The motion of a periodically vibrating surface can be “frozen” in time by pulsing the illumination in synchrony with the oscillation frequency of the sample. This technique requires prior knowledge of the oscillation frequency and light pulses whose duration is a small fraction of the vibration period.

In many applications the vibration is excited at a known frequency by an external drive signal. By introducing a controlled phase delay between the illumination pulses and this drive signal, a specific phase instant of the oscillation can be isolated. Repeating the measurement at multiple phase delays spanning a full oscillation period then allows the complete vibrational motion of the object to be reconstructed.

Two important considerations arise when capturing stroboscopically illuminated interferometric images:

(i) **Motion blur** If the light-pulse duration is not short compared with the vibration period, temporal averaging within each pulse causes motion blur, leading to a systematic underestimation of the vibration amplitude.

(ii) **Pulse-count variation** Because each camera frame integrates light from multiple illumination pulses, the driving frequency and the camera frame rate must be chosen such that their ratio avoids near-integer values. Otherwise, the number of pulses captured per frame can fluctuate between successive exposures, producing intensity variations in the recorded image sequence. This effect becomes particularly pronounced when the number of pulses per exposure is small—for instance, at low pulse-repetition frequencies or short exposure times.

2.2.2 Discrete Fourier Transform

Many measurement, digital-signal-processing, and control applications require the analysis of periodic signals. In such applications the data is typically sampled at discrete, equidistant time points and is of finite length. A data set consisting of N samples can be represented as an $N \times 1$ (complex-valued)

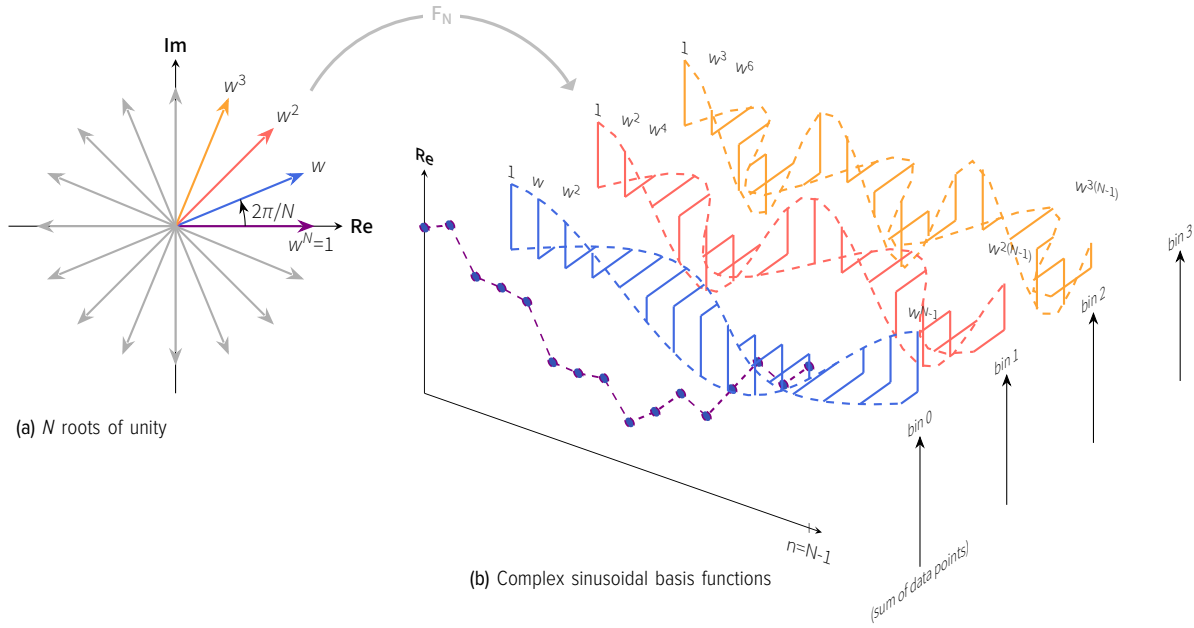


Figure 2.4 | Discrete Fourier Transform and its complex exponential basis. The DFT is constructed from the primitive N th root of unity $w = \exp(j2\pi/N)$ and its integer powers. Panel (a) illustrates the N roots of unity as equally spaced points on the unit circle in the complex plane. For a given bin index k , the factors w^{kn} , $n=0, \dots, N-1$, form a discrete complex sinusoid with a constant phase increment of $2\pi k/N$ between successive samples. Panel (b) shows how these sequences constitute the rows (or, depending on convention, the columns) of the Fourier matrix $\mathbf{F}_N$ (Eq. 2.15). The DFT coefficients are obtained by projecting the sampled signal onto these complex basis functions (Eq. 2.16); each bin therefore measures the contribution of one discrete frequency component to the signal. The $k=0$ basis function is constant, since $w^{0n}=1$ for all n , and the corresponding DFT coefficient equals the sum of all signal samples, i.e., the DC component (when divided by M).

vector

$$\mathbf{y} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}. \quad (2.11)$$

The Discrete Fourier Transform (DFT) of $\mathbf{y}$ yields Fourier coefficients—often referred to as DFT-bins—defined as

$$c_k = \sum_{n=0}^{N-1} y_n e^{-j\frac{2\pi}{N}kn}, \quad k = 0, 1, \dots, N-1. \quad (2.12)$$

Introducing the N th root of unity (Fig. 2.4a)

$$w := e^{j\frac{2\pi}{N}}, \quad (2.13)$$

Eq. (2.12) can be rewritten as

$$c_k = \sum_{n=0}^{N-1} y_n w^{-kn}, \quad k = 0, 1, \dots, N-1. \quad (2.14)$$

Collecting all N equation into the $N \times N$ Fourier matrix

$$\mathbf{F}_N = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w & w^2 & \dots & w^{N-1} \\ 1 & w^2 & w^4 & \dots & w^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w^{N-1} & w^{2(N-1)} & \dots & w^{(N-1)^2} \end{bmatrix}, \quad (2.15)$$

the DFT admits the compact matrix form

$$\mathbf{c} = \mathbf{F}_N^* \mathbf{y} \quad (2.16)$$

Because $\mathbf{F}_N \mathbf{F}_N^* = N\mathbf{I}$ (see Ref. [20] for a proof), $\mathbf{F}_N$ is invertible and the DFT therefore constitutes a unique, bijective linear map.

In essence, the DFT projects $\mathbf{y}$ onto a set of N complex discrete sinusoidal basis functions of frequencies k/N , with $k = 0, 1, \dots, N-1$ (Fig. 2.4b). Hence, each bin—a complex number—represents a measure of how well $\mathbf{y}$ resembles a sinusoid of the respective frequency. This gives rise to two important considerations regarding the data in $\mathbf{y}$:

(i) Nyquist limit The DFT can only resolve frequencies up to half the sampling frequency (Nyquist–Shannon sampling theorem). For any purely real-valued $\mathbf{y}$, every coefficient c_k shares a complex-conjugate partner c_{N-k} , so that only the first $\lfloor N/2 \rfloor + 1$ bins carry independent information. The amplitude and phase of each spectral component can be extracted from

$$\mathbf{C} = \frac{2}{N} \mathbf{F}_N^* \mathbf{y}, \quad (2.17)$$

by computing the magnitude $|C_k|$ and argument $\arg(C_k)$ of the respective bin.

(ii) Spectral leakage The DFT implicitly assumes that $\mathbf{y}$ contains an integer number of periods of every constituent sinusoid. When this condition is not satisfied, energy from a single spectral component spreads across multiple adjacent bins—a phenomenon known as spectral leakage.

2.2.3 Shot Noise

Shot noise, also known as Poisson noise, is a fundamental form of uncertainty associated with the measurement of discrete, quantised events. It arises whenever the underlying process can be modelled by a Poisson distribution—that is, when outcomes are non-negative integers, events occur independently of one another, and the probability of more than one event in an infinitesimally small interval is negligible.

In the context of this work, the Poisson model applies to image sensors, which measure scene irradiance by counting the number of discrete photons incident on each pixel during a given exposure interval.⁶ Because photon arrivals are governed by quantum-mechanical statistics and are therefore inherently random, the resulting fluctuations in the measured count constitute shot noise.

The probability of recording exactly k photon events when the expected (mean) count is $\langle k \rangle = \lambda$ is given by the Poisson probability mass function (Fig. 2.5a):

$$p(k) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad k = 0, 1, 2, \dots \quad (2.18)$$

A distinctive property of the Poisson distribution—in contrast to the Gaussian distribution, where the mean and variance are independent parameters—is that it is fully characterised by the single parameter λ : the variance equals the mean. This can be shown as follows. The variance is defined as

$$\sigma^2 = \langle (k - \langle k \rangle)^2 \rangle = \langle k^2 \rangle - \langle k \rangle^2. \quad (2.19)$$

Computing the second moment (see Ref [21] for a full derivation):

$$\langle k^2 \rangle = \sum_{k=0}^{\infty} k^2 \frac{\lambda^k e^{-\lambda}}{k!} = \dots = \lambda^2 + \lambda. \quad (2.20)$$

⁶In digital sensors, the photoelectric effect converts incident photons into electron–hole pairs, whereas film-based sensors rely on photosensitive chemical reactions.

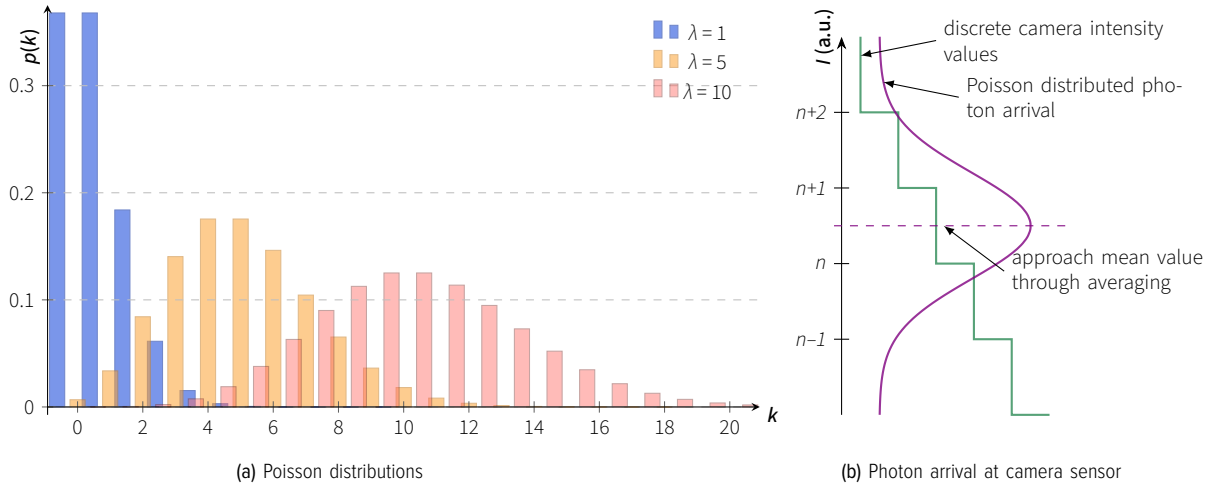


Figure 2.5 | Poisson statistics and discrete intensity detection. The left panel (a) shows Poisson distributions for different mean event numbers λ . As λ increases, the distribution broadens and becomes more symmetric; for sufficiently large λ , it is well approximated by a Gaussian distribution with mean λ and variance λ , although the underlying detection statistics remain Poissonian. The right panel (b) schematically illustrates the detection of photon-induced signal levels by a quantized camera pixel. If the statistical fluctuations of the detected signal span multiple quantization levels, repeated independent measurements can be averaged to estimate the underlying mean intensity with sub-quantization precision, i.e., with a resolution finer than the least significant bit of a single recorded sample.

Substituting into Eq. (2.19) gives

$$\sigma^2 = \lambda^2 + \lambda - \lambda^2 = \lambda. \quad (2.21)$$

and therefore the standard deviation is

$$\sigma = \sqrt{\lambda}. \quad (2.22)$$

Shot noise is thus signal-dependent: its standard deviation equals the square root of the mean signal. The signal-to-noise ratio (SNR) due to shot noise alone is

$$\text{SNR} = \frac{\lambda}{\sigma} = \frac{\lambda}{\sqrt{\lambda}} = \sqrt{\lambda} \quad (2.23)$$

Thus, increasing the number of detected photons improves the SNR, but only with the square root of the signal. For example, doubling the SNR requires four times as many detected photons.

In imaging systems, a larger value of λ can be achieved by increasing the exposure time, increasing the illumination level, improving the collection efficiency of the optical system, or averaging repeated measurements. When averaging M statistically independent images of the same scene, the noise standard deviation is reduced by a factor of $\sqrt{M}$, and the SNR improves by the same factor.

Depending on the illumination conditions and the sensor bit depth, shot noise can cause the detected photon count to fluctuate across multiple digitisation levels. If repeated measurements are available and the sensor response is sufficiently stable and calibrated, this stochastic variation effectively provides sub-bit-depth information: by averaging over many frames, measurement precision can exceed the quantisation limit imposed by the fixed bit depth of the analogue-to-digital converter (Fig. 2.5b). This principle is closely related to the concept of dithering in signal processing.

In practice, shot noise is only one of several relevant sensor-noise sources. Additional contributions may arise from read noise, dark-current noise, thermal noise, quantization noise, and fixed-pattern noise.

2.3 Summary and Further Reading

This chapter introduced the theoretical foundations underlying the full-field interferometric measurement system described in Chapter 3. On the optical side, it covered key concepts such as the superposition of electromagnetic waves, the operating principles of interferometry, and the coherence properties of light sources. These concepts are pivotal for understanding the formation and interpretation of interference signals. For a contemporary approach to optics, Hecht's work [22] is recommended. Additionally, Hariharan's [23, 24] and Goodman's [25] texts serve as standard references on optical coherence and interferometry. For optical measurements specific to MEMS characterization, Osten et al. [26] provide detailed insights.

Regarding data acquisition and processing for vibration measurements, this chapter outlined the principles of stroboscopic interferometric detection, the discrete Fourier transform (DFT) serving as a spectral analysis tool for periodic measurement data, and the statistical origins of shot noise. The detailed discussions on stroboscopic and time-averaged optical methods for vibration analysis are presented in Powell and Stetson's study [27]. Broader treatments of optical metrology and interferometric measurement techniques are also available in the work by Osten et al. [28]. The DFT derivation closely follows the approach of Meyberg and Vachenauer [20], where a derivation of the fast Fourier transform (FFT) algorithm is also available. More in-depth discussions of the DFT and its interpretation can be found in the work by Oppenheim and Schaffer [29]. For a thorough understanding of statistics covering Poisson-distributed events, [21] is a standard reference. In the realm of optical detection and imaging, shot noise is also comprehensively discussed by Goodman [25] and Janesick [30].

3 Measurement System Development

3.1 System Architecture

3.1.1 System Components

The measurement system comprises a Michelson-type interferometer setup similar to the one presented in Ref. [16, 17]. Illumination is provided by a red LED with center wavelength $\lambda_0 \approx 624 \text{ nm}$ and full width at half maximum (FWHM) spectral width $\Delta\lambda \approx 18 \text{ nm}$ [31]. The LED beam is split into reference and sample arms by a non-polarizing beam-splitter cube. After reflection from a reference mirror and from the device under test (DUT), the beams recombine at the splitter and form an interference pattern that is recorded by a digital monochrome camera (Fig. 3.1a).

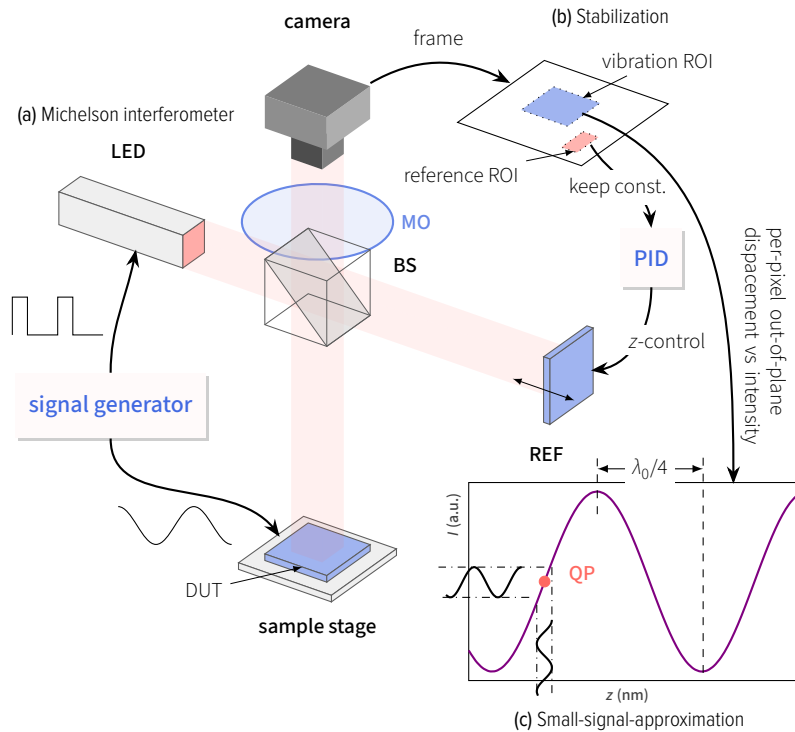


Figure 3.1 | Experimental setup and operating principle. (a) **Michelson interferometer:** LED illumination is split by a non-polarizing beam splitter (BS) into reference and sample arms. The reference mirror is mounted on a piezo z-stage (REF); the sample is imaged through magnification optics (MO) onto the camera. The signal generator provides synchronized LED pulses and a sinusoidal excitation of the DUT (stroboscopic illumination). (b) **Stabilization:** a non-vibrating *reference ROI* provides a mean intensity used as an error signal for a PID controller that drives the piezo to maintain the operating point at the chosen quadrature point (QP) of the *vibration ROI*. (c) **Small-signal approximation:** intensity I vs. displacement z is linearized at the QP, where the local slope $s = dI/d\phi$ is maximal. The per-pixel displacement-to-intensity gain $G = (4\pi/\lambda_0)s$ maps measured intensity modulation in the *vibration ROI* to out-of-plane displacement.

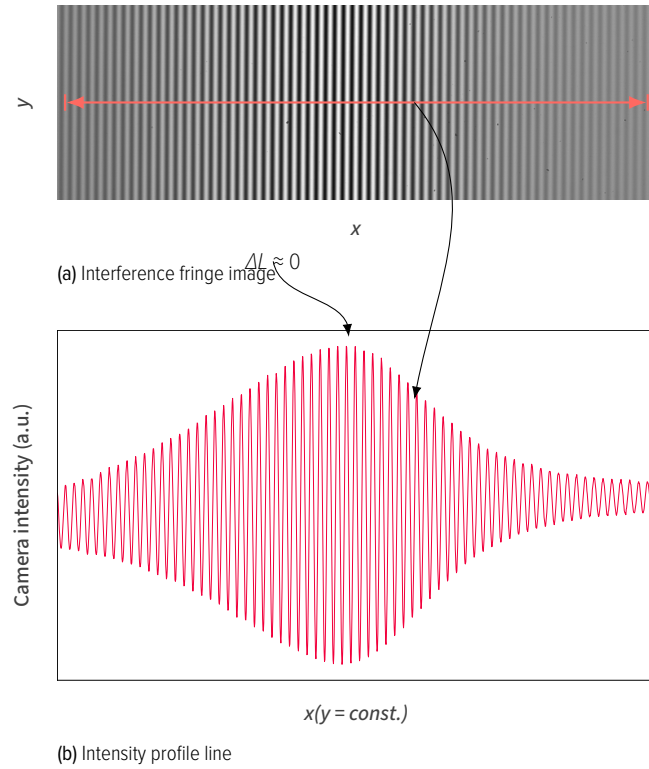


Figure 3.2 | Visualization of LED coherence envelope. Tilting a reflective sample in the xz -plane introduces a continuous path-length gradient along the x -direction, producing vertical interference fringes in the reflection image (a). As the path-length mismatch increases, the fringe contrast decays according to the source's coherence envelope, visible in the extracted intensity profile (b).

To avoid speckle that can degrade spatial resolution, the system uses LED illumination. The LED's low spatial coherence produces speckle-free images, allowing the camera's native resolution to be fully utilized. At the same time, the LED provides sufficient temporal coherence for interferometric detection within the intended path-length mismatch range. A simple way to visualize this is to introduce a controlled path-length mismatch (e.g., by a slight mirror tilt) and inspect the fringe visibility across the field: the visibility decreases with increasing mismatch according to the coherence envelope (Fig. 3.2). For reference, the LED's coherence length can be approximated by $l_c \simeq \lambda_0^2 / (2\Delta\lambda)$ (see Sec. 2.1.3), implying tens of micrometers for the given parameters.¹

Coarse displacement of the reference mirror along the optical z -axis is provided by a translation stage, while fine positioning is achieved with a piezoelectric actuator driven by a voltage controller. A single signal generator drives both the LED and the DUT. For stroboscopic measurements, the generator produces short voltage pulses to the LED and simultaneously applies a sinusoidal excitation to the DUT at a synchronized frequency. In the current configuration, the signal generator allows for stroboscopic measurements at up to the low-tens of megahertz.

All instruments—the piezo controller, camera, and signal generator—are controlled by custom software (Ch. 5).

3.1.2 Stabilization

Over the duration of a measurement run, the interferometer operating point can drift due to mechanical creep, thermal expansion, or sudden disturbances, which translates into unwanted optical path-length changes and measurement error. Continuous stabilization is therefore required.

¹Envelope effects are explicitly considered in Sec. 3.1.3.

The system implements software-based stabilization using a PID controller that derives feedback directly from the camera image (Fig. 3.1b). Prior to measurement, a non-vibrating *reference region* is selected on the sample. Its mean intensity is continuously compared to a setpoint, and the PID loop drives the piezo actuator to maintain the operating point. A notable constraint is that the achievable closed-loop bandwidth is limited by the camera frame rate. For increasing frame rates, the processing latency of the acquisition pipeline eventually sets the bottleneck for the stabilization speed.

3.1.3 Operating Principle

For two-beam interference with (temporally) partially coherent illumination, the recorded intensity at a pixel can be written as

$$I(t) = I_r + I_s + 2\sqrt{I_r I_s} \gamma(\Delta L) \cos[\Delta\varphi(t)], \quad (3.1)$$

where I_r and I_s are the reference and sample arm intensities at the detector, $\gamma(\Delta L)$ describes the degree of temporal coherence (envelope) as a function of path-length mismatch ΔL , and $\Delta\varphi(t)$ is the phase difference. If the phase excursion at a pixel remains well within a single fringe near the maximum of the coherence envelope, the envelope-induced amplitude modulation can be treated as approximately constant (e.g. γ_0) and Eq. (3.1) can locally be described by an offset cosine:

$$I(t) \approx I_r + I_s + 2\sqrt{I_r I_s} \gamma_0 \cos[\Delta\varphi(t)]. \quad (3.2)$$

For normal incidence with reflection at the sample and out-of-plane displacement $z(t)$,

$$\Delta\varphi(t) = \Delta\varphi_0 + \frac{4\pi}{\lambda_0} z(t), \quad (3.3)$$

where $\Delta\varphi_0$ is the static operating point phase. Linearizing around $\Delta\varphi_0$ gives

$$I(t) \approx I(\Delta\varphi_0) + \left. \frac{dI}{d\varphi} \right|_{\Delta\varphi_0} \frac{4\pi}{\lambda_0} z(t). \quad (3.4)$$

Mapping camera pixels to (x, y) , define the *pixel sensitivity* (slope with respect to phase) as

$$s(x, y) = \left. \frac{dI}{d\varphi}(x, y) \right|_{\Delta\varphi_0(x, y)}. \quad (3.5)$$

For convenience, the *displacement-to-intensity gain* is also defined

$$G(x, y) = \frac{4\pi}{\lambda_0} s(x, y), \quad (3.6)$$

so that the small-signal model reads $I(t) \approx I(\Delta\varphi_0) + G(x, y) z(t)$. The surface displacement $z(t)$ around the operating point to intensity mapping is thus:

$$z(t) = \frac{\Delta I(t)}{G(x, y)}, \quad (3.7)$$

where $\Delta I(t) = I(t) - I(\Delta\varphi_0)$. The small-signal approximation is most sensitive when the operating point is at quadrature (QP),

$$\Delta\varphi_0 = \frac{(2m+1)\pi}{2}, \quad m \in \mathbb{Z}, \quad (3.8)$$

where $|dI/d\varphi|$ is maximal. In this regime, sub-(quarter-)wavelength out-of-plane displacements can be converted into intensity variations via the known local slope (Fig. 3.1c). The approximation requires

that z remains small enough for the linear model to hold since nonlinearity introduces error as z grows.²

Because the camera provides full-field imaging, each pixel serves as an independent measurement channel. A spatial vibration map is thus acquired simultaneously in a single measurement run.

3.2 Measurement Procedure

3.2.1 Quadrature Point Detection (Calibration)

Before vibration measurements, the stabilization setpoint must be determined. A calibration sweep is performed by scanning the piezo-driven z stage while recording interference images with the DUT electrically inactive (to avoid motion-induced modulation). For each frame z , compute (i) the mean intensity μ_{ref} over a *reference region* (assumed non-vibrating) and (ii) the mean intensity μ_{vib} over the *vibration region* (the area of interest). Because the vibration region is generally not perfectly flat, it is impossible to stabilize all pixels there exactly at quadrature. Averaging over this region during the sweep yields an operating point that is near-optimal for the entire vibration region.

Depending on the average relative z -offset of the reference and vibration regions, the curves $\mu_{\text{ref}}(z)$ and $\mu_{\text{vib}}(z)$ are phase-shifted relative to one another. Sinusoidal models

$$\mu_{\text{ref}}(z) \approx a_r + b_r \cos(kz + \phi_r), \quad (3.9)$$

$$\mu_{\text{vib}}(z) \approx a_v + b_v \cos(kz + \phi_v), \quad (3.10)$$

can be fitted within a span of $\leq$ one fringe (see Eq. (3.2)). The QP positions are given by $kz + \phi_v = (2m+1)\pi/2$. The *reference intensity setpoint(s)* for the PID loop are then obtained by evaluating $\mu_{\text{ref}}(z)$ at these z values:

$$I_{\text{set}}^{(m)} = \mu_{\text{ref}}\left(z : kz + \phi_v = \frac{(2m+1)\pi}{2}\right). \quad (3.11)$$

Illumination conditions must remain unchanged between calibration and measurement to preserve the intensity–displacement mapping. Fig. 3.3 shows a calibration fit of real measurement data.

To obtain the operating-point slope per pixel during measurement, all frames within the fit window used above are stored. Later, a per-pixel fit of the same form

$$I_{x,y}(z) \approx a_{x,y} + b_{x,y} \cos(kz + \phi_{x,y}) \quad (3.12)$$

is performed over those frames. Since

$$\frac{dI}{dz}(x,y) = \frac{dI}{d\phi}(x,y) \frac{d\phi}{dz}(x,y), \quad (3.13)$$

the *pixel sensitivity* at the selected operating point z^* is

$$s(x,y) = \left. \frac{dI_{x,y}}{d\phi} \right|_{z^*} = - \frac{b_{x,y} \sin(kz^* + \phi_{x,y})}{k}, \quad (3.14)$$

and the displacement-to-intensity gain follows from Eq. (3.6). This yields a “sensitivity” (gain) map with which each pixel’s intensity time series during the measurement is converted to vibration amplitude.

²At a quadrature point, linearization errors below 5% follow $\frac{z}{\sin(z)} < 1.05$. This corresponds to deviations of approx. $\lambda_0/100$ (~ 6 nm).

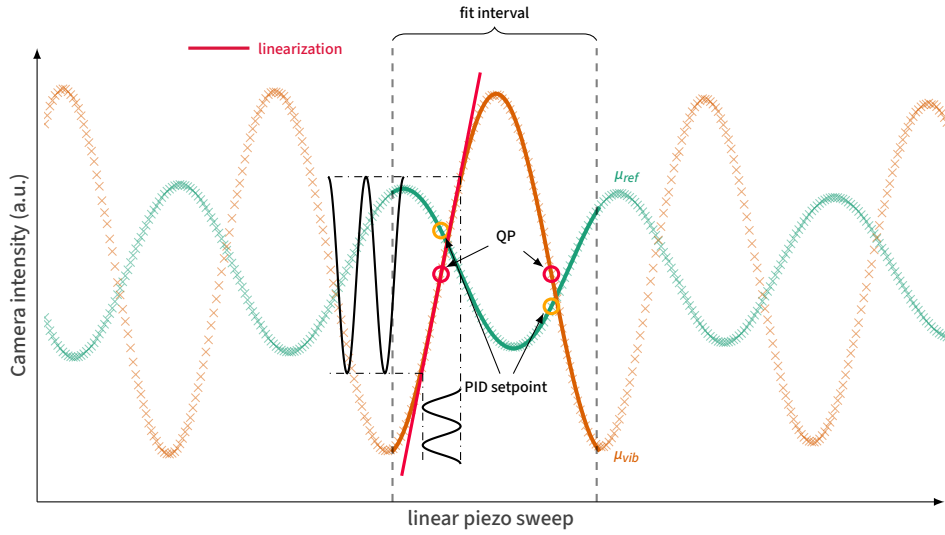


Figure 3.3 | Calibration sweep and operating-point selection from representative measurement data. Mean intensities of the reference ROI (μ_{ref}) and vibration ROI (μ_{vib}) are plotted versus the linear piezo z-sweep. One clean fringe is used to fit sinusoids, from which quadrature points (QP) are identified on μ_{vib} and the corresponding PID setpoint(s) are read off on μ_{ref} . The local linearization at the selected operating point yields the sensitivity (Eqs. 3.13–3.14).

Quadrature Point Selection Considerations Modulation depth b_v of $\mu_{vib}(z)$ directly impacts sensitivity. Larger b_v increases the local slope magnitude at the operating point, so a given displacement maps to a larger range of digital counts, improving SNR. Consequently, camera exposure and gain and LED pulse parameters (voltage, duty cycle) should be tuned to maximize modulation depth while avoiding saturation or clipping, which would bias the fit and the inferred slope.

The relative z-offset between the reference and vibration regions also determines how close the chosen setpoint lies to extrema of $\mu_{ref}(z)$. If drift pushes the operating point beyond an extremum, the slope changes sign and the controller can relock to a neighboring quadrature point one fringe away. Selecting a setpoint sufficiently far from the extrema reduces this risk and improves robustness during long acquisitions.

3.2.2 Measurement Principle and Parameters

Two main data-acquisition modalities are implemented, illustrated in Fig. 3.4: (i) phase stepping with frame averaging and (ii) continuous frame acquisition with a slight frequency shift between excitation and illumination.

Phase stepping and averaging Driving the LED and the DUT at the same frequency ($f_{LED} = f_{DUT}$), a static camera image at well defined phase positions can be obtained by applying a phase delay $\theta_{LED} - \theta_{DUT}$ to the driving signals. One vibration cycle is reconstructed by stepping the phase over 2π in M increments ($\Delta\theta = 2\pi/M$).

Random quantization errors from shot noise and read noise—both dependent on exposure, gain, and illumination—are reduced by averaging N frames per phase step, improving SNR approximately as $\sqrt{N}$. Theoretically, this averaging can improve accuracy beyond the fixed camera bit depth³ (see Sec. 2.2.3).

A practical advantage of this modality is its robustness to occasional disturbances. Frames need not be acquired strictly consecutively: using the PID error signal (Sec. 3.1.2), frames taken during large

³The present system employs a 12 bit camera sensor

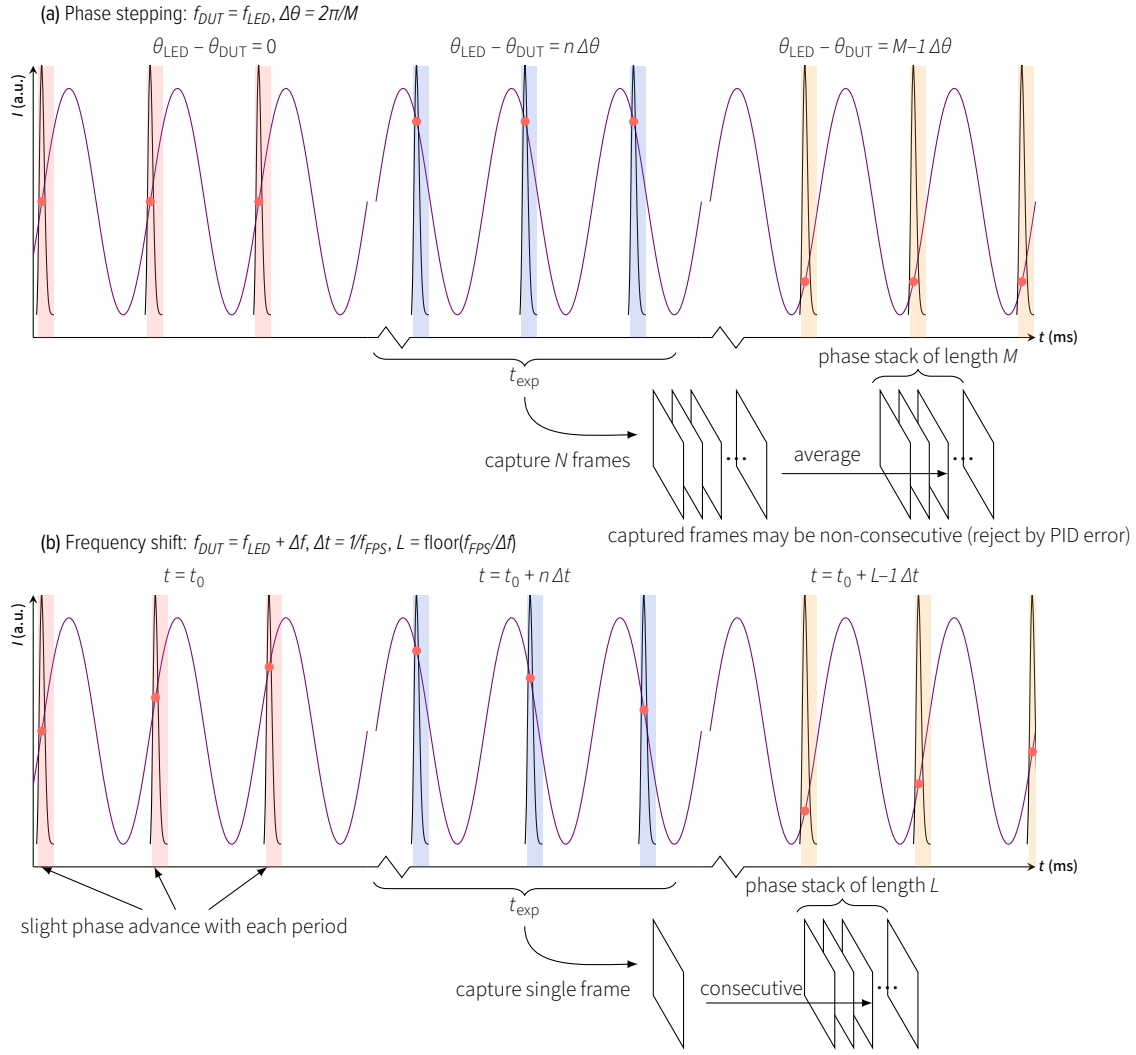


Figure 3.4 | Data acquisition modalities. (a) **Phase stepping with averaging** (a): the LED and DUT are driven at the same frequency ($f_{DUT} = f_{LED}$). Frames are acquired at M discrete, commanded phase positions ($\Delta\theta = 2\pi/M$). For each phase delay $n\Delta\theta$, N frames (not necessarily consecutive in time) are averaged improving the signal-to-noise ratio (SNR). (b) **Frequency-shift (detuned) acquisition** (b): the LED strobe is detuned by Δf from the DUT ($f_{DUT} = f_{LED} + \Delta f$), so successive frames (spaced by $\Delta t = 1/f_{FPS}$) sample the vibration at a constant phase increment over the beat period $T_{beat} = 1/\Delta f$, producing an apparent slow sweep enabling rapid qualitative visualization of the vibration field at a target frequency.

deviations from the setpoint can be flagged as low quality and excluded from the average for their respective phase step. This allows reliable phase-stepped reconstruction even under imperfect stability conditions.

Continuous frame acquisition with shifted frequencies An alternative modality drives the LED and DUT at slightly different frequencies, i.e. $f_{DUT} = f_{LED} + \Delta f$. In this case, the relative phase evolves at the beat frequency Δf , causing the camera to observe the high-frequency DUT motion in apparent slow motion over a synthetic period $T_{beat} = 1/\Delta f$. This approach enables rapid scanning across large frequency ranges and provides immediate visual feedback of the vibration field at f_{DUT} directly in the live camera feed.

However, this modality is more sensitive to timing irregularities. Frame drops or small deviations in inter-frame timing—typically induced by software latency—accumulate over the synthetic slow-motion cycle and can decorrelate the nominal phase index across consecutive cycles. This complicates coherent averaging and limits quantitative reconstruction. Consequently, while very useful for fast qualitative assessment (e.g., identifying resonance peaks or verifying modal shapes), the absence of guaranteed frame lock in the current system constrains the quantitative accuracy of this measurement

mode.

3.2.3 Error Sources and Mitigation

The system directly addresses two dominant error mechanisms: (i) operating-point drift along z via the camera-based PID stabilization (Sec. 3.1.2) and (ii) intensity noise via frame averaging. Additional sources and mitigations are as follows.

Piezo nonlinearity and hysteresis Piezo actuators exhibit drive nonlinearity and hysteresis. During the calibration sweep, a linear piezo-voltage ramp can yield slightly non-uniform fringe spacing in $I(z)$. However, when analysis is restricted to a single clean fringe within the coherence envelope, a sinusoidal fit aligns well with the data (Fig. 3.3), rendering the nonlinearity negligible in the local region relevant to measurement. Because the calibration ramp is unidirectional, hysteresis does not bias the fit. During measurement the PID applies small, bidirectional corrections around the setpoint; no adverse effects were observed.

Camera sensor temperature dependence Typical industrial cameras lack active sensor cooling. As the sensor warms up under continuous acquisition, temperature-dependent dark current and related offsets can drift, leading to inconsistent intensity levels between the beginning and end of a long run. In the present system, this is mitigated by (i) a short warm-up period prior to frame-acquisition and (ii) a secondary control loop that adjusts camera gain to maintain constant intensity in a region that does not exhibit interference (i.e., a surface that has significant z -offset (ten of micrometers) from the vibration ROI). Functionally, this acts as a PID on the camera, using an out-of-interference ROI as the reference signal⁴.

Lateral creep While the z -axis stabilization handles optical path-length drift, the current setup does not actively correct lateral (in-plane) creep. Over long runs (several hours), small (a few micrometer) relative shifts among camera, lens, and sample can accumulate, affecting pixel-wise correspondence and long averaging sequences. In the present work, this has not yet been addressed. Potential remedies include mechanical/thermal stabilization, and/or software-based drift detection (e.g. fft phase correlation) to correct ROI coordinates or possibly drive a motorized repositioning subsystem to maintain alignment during extended measurements.

3.3 Summary

The presented full-field interferometric measurement system based on a Michelson-type interferometer counteracts operating-point drift by employing a software PID loop that derives its error signal directly from a non-vibrating reference region in the camera image. Operating principles were formalized starting from the two-beam interference model, leading to a small-signal linearization around a local operating point. The corresponding pixel-wise slope (sensitivity) and displacement-to-intensity gain provide a calibrated mapping from measured intensity variations to out-of-plane displacement.

A calibration procedure determines the stabilization setpoint(s) and the per-pixel sensitivity (gain) map. The piezo z -stage is swept over (at least) one clean fringe while the DUT is inactive; sinusoids are fit to mean intensities in reference and vibration regions to locate quadrature points and to set the stabilization target. Storing the same frames enables per-pixel sinusoidal fits, from which the local slope and gain are computed at the selected operating point.

Two acquisition modalities were implemented. In *phase stepping with averaging*, the LED pulses are

⁴intensity setpoint is the mean value measured during calibration in the same ROI

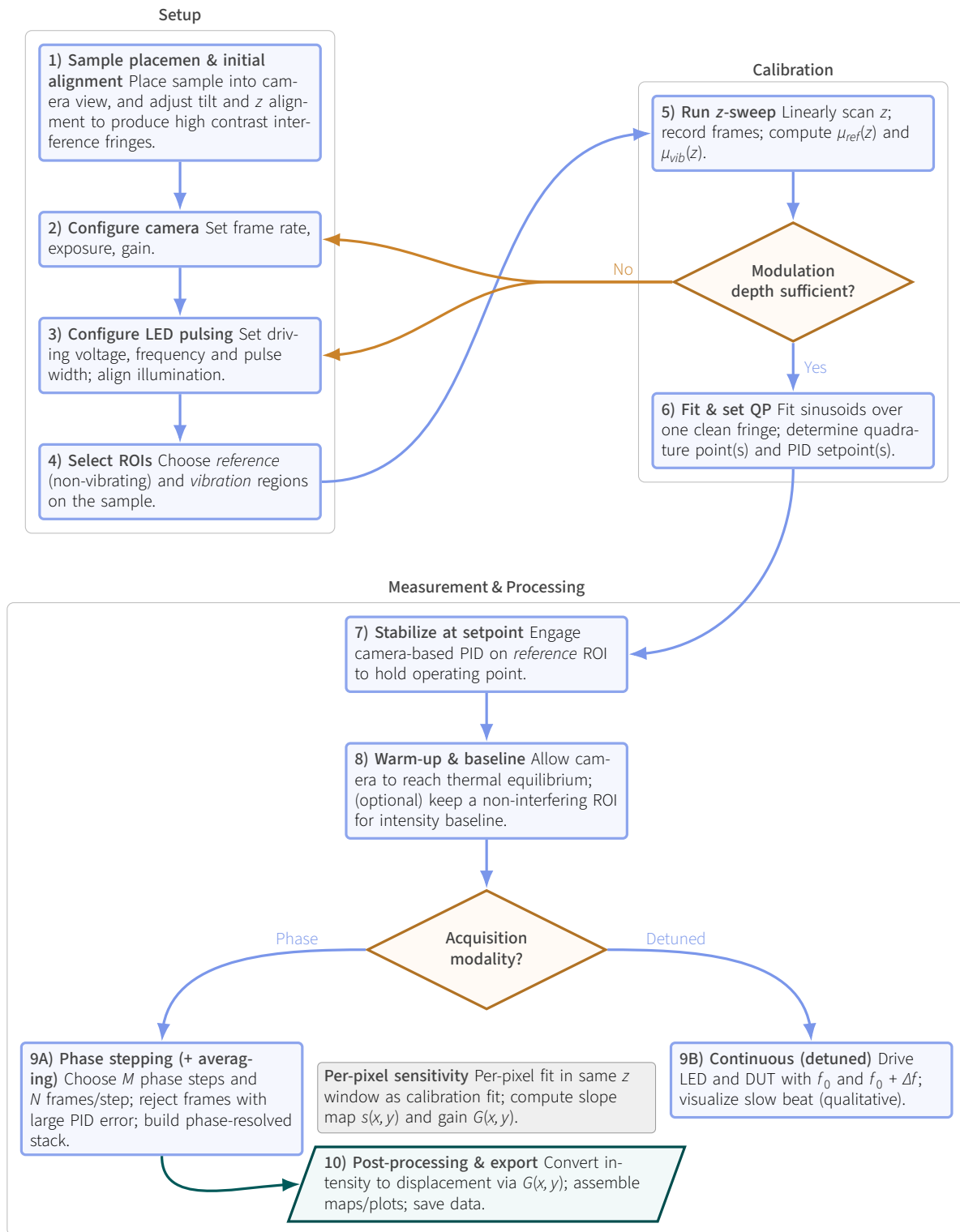


Figure 3.5 | Measurement workflow. Setup, calibration (with modulation-depth check and sensitivity-map generation), and measurement/processing with two acquisition modalities.

phase-stepped relative to the DUT excitation to reconstruct one cycle at M phase points, with SNR improved by $\sqrt{N}$ through averaging N frames per step. Frames acquired during large stabilization excursions (identified via the PID error) can be excluded, improving robustness. In *continuous acquisition with a frequency shift*, a small detuning Δf produces a slow beat that enables near-instant visualization of high-frequency motion; it is used primarily for qualitative inspection.

Finally, additional error sources and mitigations were discussed. Piezo nonlinearity and hysteresis are

negligible within the local single-fringe calibration window. Sensor temperature drift is compensated by a brief warm-up and by a secondary gain-control loop using a non-interfering ROI. Lateral creep is currently not corrected and remains a limitation for long acquisitions; potential remedies include improved mechanical/thermal stability and software-based solutions.

An illustration of a full measurement workflow (Fig. 3.5) provides further insight into how the system components are orchestrated to deliver full-field vibration measurements.

4 Experimental Validation and Results

4.1 MEMS Cantilever Sample

This chapter validates the interferometric measurement system on a representative MEMS device: an array of piezoelectrically actuated cantilever beams fabricated on a single chip. All cantilevers share nominally identical geometry and material stack. A micrograph of the chip is shown in Fig. 4.1a. The device consists of a substrate with a piezoelectric layer and electrodes; anisotropic backside etching releases the cantilevers as suspended beams. The exact material composition and fabrication specifics are undisclosed and not required for the analyses presented here.

With the current magnification, the spatial sampling is roughly $1\text{ }\mu\text{m}$ per pixel. Individual beams span ~ 400 pixels in length and ~ 80 pixels in width. All beams are driven simultaneously via the integrated piezoelectric layer, producing out-of-plane motion under an applied AC voltage.

Because cantilevers are well-characterized, their frequency response provides a principled basis for validating the measurement data against analytical predictions. In particular, Euler–Bernoulli theory offers closed-form mode shapes and natural frequency scalings for clamped–free beams that serve as useful references. Beyond frequency response, the effect of frame averaging on measurement sensitivity is also qualitatively investigated.

Due to fabrication tolerances, cantilevers may exhibit slight static curvature or tilt relative to the wafer plane. Since the interferometric system performs best on locally flat surfaces (Sec. 3.1.3), a device with minimal out-of-plane static deformation was selected and aligned the beam’s mean surface normal with the optical axis. The following analysis is restricted to a single cantilever from the array.

4.2 Amplitude Extraction Procedure

All results in this chapter use the phase-stepping modality (Sec. 3.2.2). For a given excitation frequency, M equally spaced phase steps per vibration period are acquired. At each phase step, N frames are averaged (rejecting frames whose PID error exceeds 0.5% of the setpoint), and this phase sequence is repeated over n_p periods to increase robustness to occasional outliers. The raw acquisition produces a 4D stack $Y \times X \times M \times n_p$ (height, width, phase index, period index). After averaging across periods, a phase stack $Y \times X \times M$ is obtained.

At each pixel (x, y) , the intensity as a function of phase step index m (with phase $\phi_m = 2\pi m/M$) is modeled as

$$I_{x,y}(\phi_m) \approx I_0(x, y) + A_1(x, y) \cos[\phi_m - \varphi(x, y)], \quad (4.1)$$

where A_1 is the fundamental (first-harmonic) amplitude in camera units and φ is the per-pixel phase. The complex first-harmonic phasor is estimated by an M -point discrete Fourier transform (DFT) over

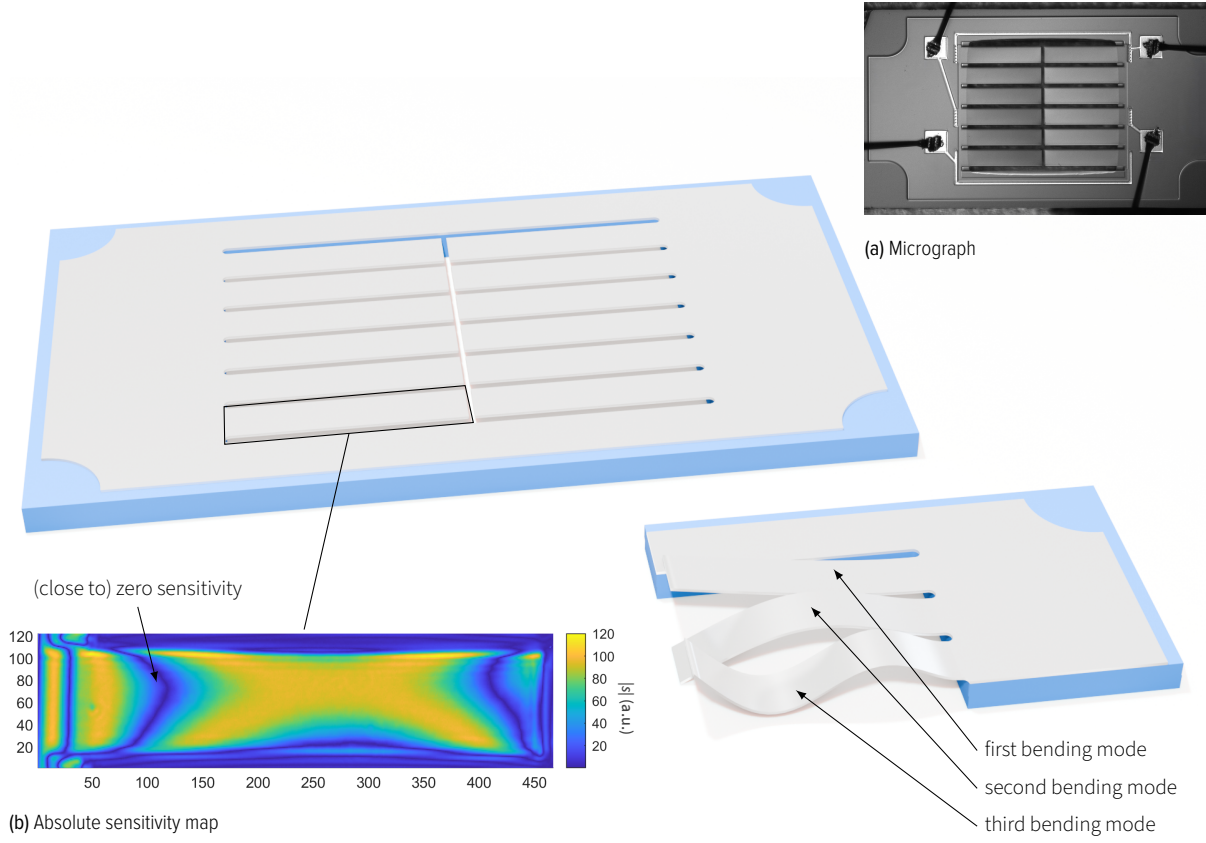


Figure 4.1 | MEMS cantilever array, micrograph, and sensitivity map. A conceptual 3D view of the piezoelectrically actuated cantilever array test chip. The first three bending modes (Euler-Bernoulli) are also illustrated. The inset **micrograph** (a) shows the wire-bonded chip. The **absolute sensitivity map** $|s(x,y)|$ obtained from a calibration sweep (b) visualizes the local small-signal slope used for amplitude mapping. Regions near the clamp and along lateral edges exhibit (near) zero sensitivity, indicating local operating points (on the beam) approaching interference extrema; non-reflective gaps between beams also appear as zero sensitivity.

phase:

$$\hat{C}_1(x, y) = \frac{2}{M} \sum_{m=0}^{M-1} I_{x,y}(\phi_m) e^{-j\phi_m}, \quad (4.2)$$

yielding

$$A_1(x, y) = |\hat{C}_1(x, y)|, \quad \varphi(x, y) = \arg\{\hat{C}_1(x, y)\}. \quad (4.3)$$

Using the per-pixel displacement-to-intensity gain $G(x, y)$ (Eq. (3.6)), obtained from the calibration fit at the operating point, the out-of-plane displacement amplitude is

$$A_z(x, y) = \frac{A_1(x, y)}{|G(x, y)|}. \quad (4.4)$$

Since sign changes of the local slope flip the displacement phase, the phase map is corrected by

$$\varphi_z(x, y) = \begin{cases} \varphi(x, y) + \pi, & \text{if } G(x, y) < 0, \\ \varphi(x, y), & \text{otherwise,} \end{cases} \quad (4.5)$$

with phase subsequently wrapped to $(-\pi, \pi]$. All results are reported as $A_z(x, y)$ (amplitude map) and $\varphi_z(x, y)$ (phase map) at the drive frequency. All measurements use LED pulsing with a 10% duty cycle.

A representative sensitivity map $s(x, y)$ (Eq. (3.5)) from the calibration sweep is shown in Fig. 4.1 4.1b. The cantilever exhibits regions of reduced or near-zero sensitivity, notably near the clamp and along

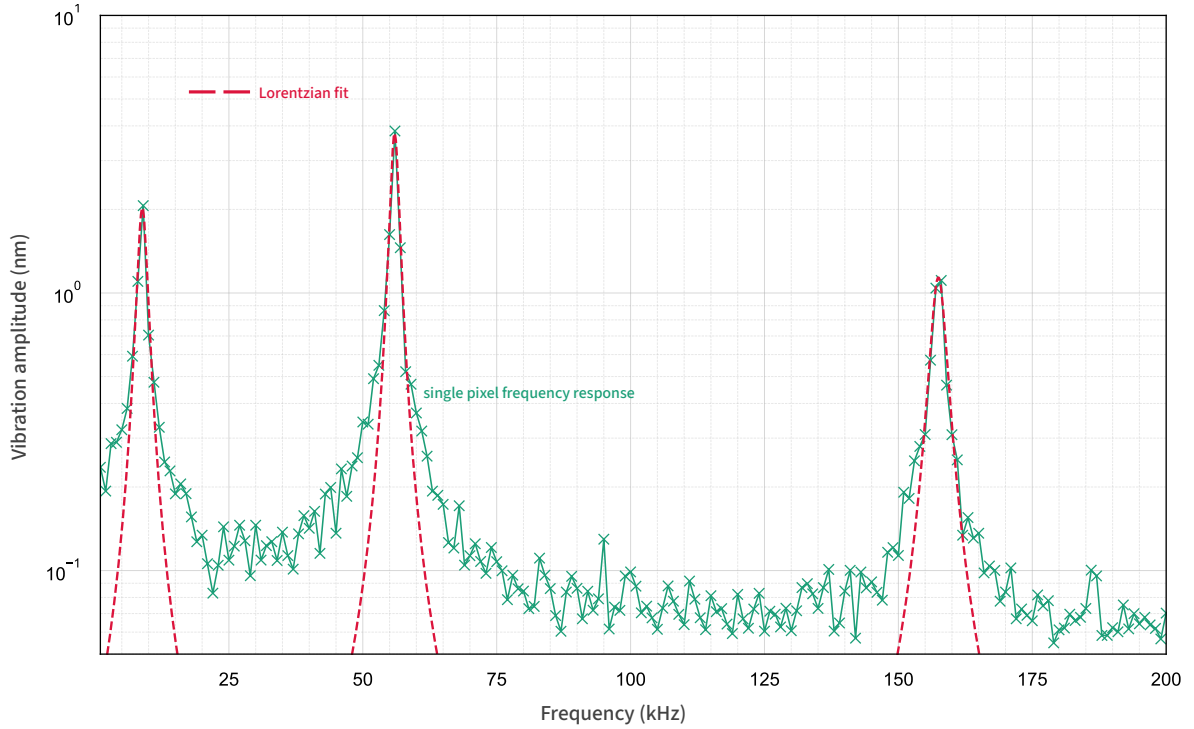


Figure 4.2 | Single-pixel frequency response and Lorentzian peak fits. Out-of-plane vibration amplitude at pixel $(x, y) = (200, 60)$ versus excitation frequency (1–200 kHz). Three pronounced resonances are observed near $f_1 \approx 9$ kHz, $f_2 \approx 56$ kHz, and $f_3 \approx 158$ kHz. The dashed red curves indicate Lorentzian fits (Eq. 4.6) used to refine peak frequencies and extract quality factors (Sec. 4.4). Amplitudes are reported in nanometers after conversion of camera units via the per-pixel gain $G(x, y)$ (Sec. 4.2).

lateral edges where slight static bending pushes the local operating point toward extrema of the interference curve. In such regions the small-signal regime is severely limited or absent, inhibiting accurate amplitude extraction. To ensure the analysis remains within the small-signal domain and to avoid division artifacts from Eq. (4.4), measurement data are masked where $|G(x, y)|$ falls below a threshold. This also improves visualization by suppressing artificially large amplitudes produced near zero sensitivity.

4.3 Frequency Response Measurement

To remain within the small-signal regime, the drive amplitude was chosen after identifying the first resonance in a preliminary sweep. The tip region (largest response in the fundamental mode) was kept well within the linearized operating range, and the same drive was used for subsequent measurements.

A frequency sweep from 1 to 200 kHz in 1 kHz steps using $M=18$ phase steps, $n_p=4$ periods, and $N=4$ frames averaged per step was performed. A pixel in the central span of the beam ($x=200$, $y=60$), sufficiently far from expected nodes of higher modes, was monitored to generate the coarse response curve (Fig. 4.2). Three prominent resonance peaks were identified at $f_1 \approx 9$ kHz, $f_2 \approx 56$ kHz, and $f_3 \approx 158$ kHz.

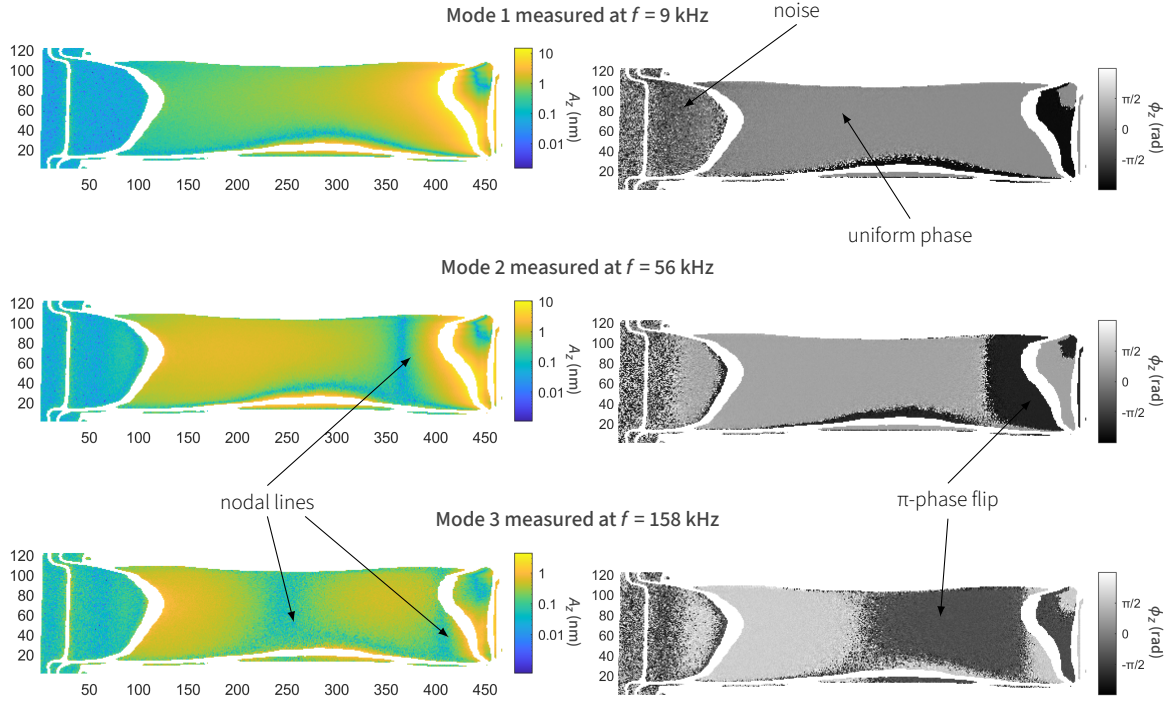


Figure 4.3 | Full-field vibration amplitude and phase at the first three bending modes. For each mode (rows), the left panel shows the out-of-plane amplitude map A_z (nm) and the right panel shows the phase map ϕ_z (rad) (phase corrected for the local slope sign and wrapped to $(-\pi, \pi]$). Nodal lines (annotated) appear as amplitude minima and coincide with π phase flips between adjacent lobes; the fundamental exhibits no internal nodes, the second mode one, and the third mode two. Regions with nearly uniform phase indicate well resolved out-of-plane displacements. Low-SNR vibrations and non-moving areas appear as noisy regions in the phase map.

4.4 Resonance Bandwidth and Quality Factor

Each resonance peak near f_n was fit with a near-resonant Lorentzian model (dashed in Fig. 4.2):

$$A(f) = A_0 \frac{1}{\sqrt{1 + \left(\frac{2Q}{f_n}\right)^2 (f - f_n)^2}}, \quad (4.6)$$

yielding refined peak frequencies $f_1 \approx 8.8$ kHz, $f_2 \approx 55.9$ kHz, and $f_3 \approx 157.5$ kHz, and quality factors $Q_1 \approx 4$, $Q_2 \approx 30$, $Q_3 \approx 48$. While these are coarse estimates, the increase of Q with mode order is consistent with reduced anchor-loss contributions and smaller effective air damping at higher frequencies.

4.5 Validation Against Euler–Bernoulli Beam Theory

To assess physical consistency, the measured resonance frequencies and mode shapes are compared with the Euler–Bernoulli model for a uniform clamped–free beam of length L :

$$f_n = \frac{\beta_n^2}{2\pi L^2} \sqrt{\frac{EI}{\rho A}}, \quad \beta_1=1.875, \beta_2=4.694, \beta_3=7.855, \dots \quad (4.7)$$

where E is Young’s modulus, I the area moment, ρ the mass density, and A the cross-section. Ratios eliminate dependence on unknown composite parameters:

$$\frac{f_n}{f_1} = \left(\frac{\beta_n}{\beta_1}\right)^2, \quad \Rightarrow \quad \frac{f_2}{f_1} \approx 6.27, \quad \frac{f_3}{f_1} \approx 17.55. \quad (4.8)$$

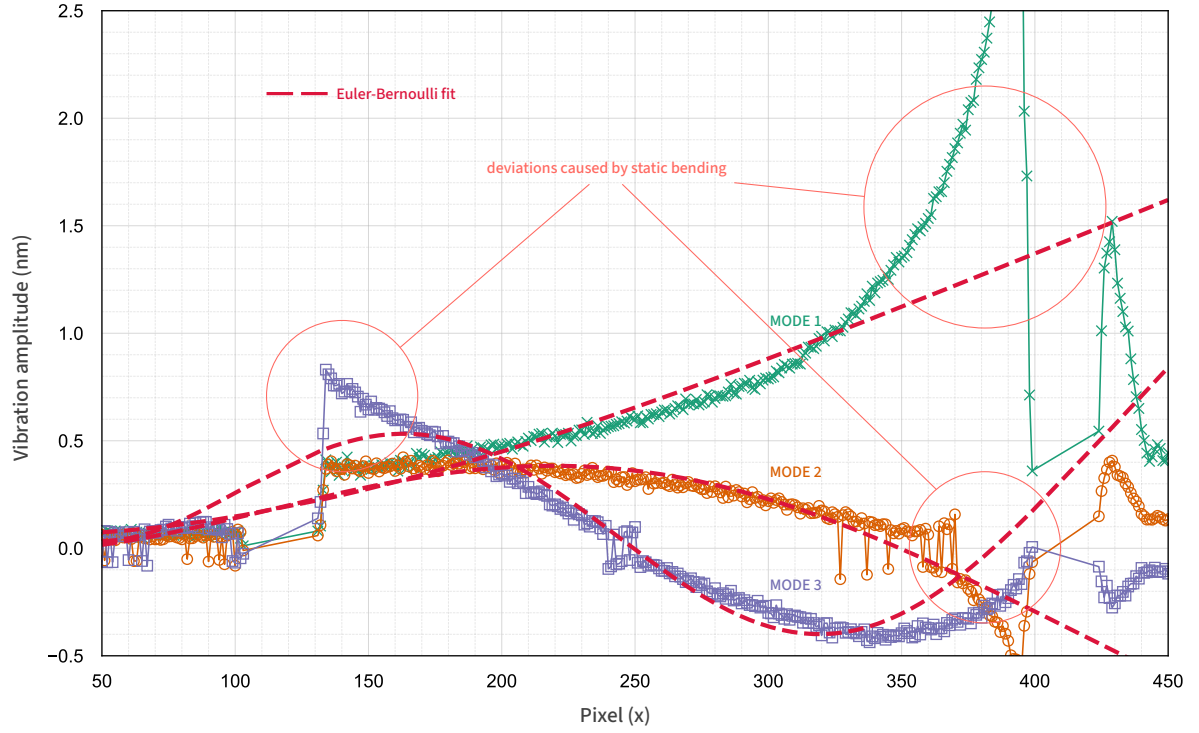


Figure 4.4 | Euler-Bernoulli mode-shape fits along the beam centerline. Measured centerline (phase corrected) amplitude profiles $A_z(x)$ at the first three bending modes compared with Euler–Bernoulli shape functions (dashed red). Node locations and lobe symmetry agree reasonably well across modes; annotated regions highlight localized deviations attributable to static bending/curvature and reduced local sensitivity near the clamp and edges. Fits were performed with an amplitude scale and effective length, consistent with the nominal ~ 400 px beam span (Sec. 4.5).

Frequency ratio comparison Table 4.1 summarizes the measured peaks and ratios normalized by f_1 , alongside the theoretical ratios from Eq. (4.8).

Mode	Measured (from fit) f_n (kHz)	Measured f_n/f_1	Theory f_n/f_1	ϵ (%)
1	8.8	1.0	1.00	0
2	55.9	6.3	6.27	1.0
3	157.5	17.8	17.55	1.6

Table 4.1 | Measured resonance frequencies and ratios vs. Euler–Bernoulli predictions.

The good degree of agreement, quantified as $\epsilon_n = |(r_n - r_n^{\text{th}})/r_n^{\text{th}}| \times 100\%$ validates the physical consistency of the obtained mode frequency values.

Mode shape consistency Phase maps at f_1 , f_2 , and f_3 (Fig. 4.3) exhibit the expected nodal structure of clamped–free bending: zero internal nodes for the fundamental, one node for the second mode, and two for the third, with symmetry consistent with Euler–Bernoulli shapes $\phi_n(x)$ along the beam centerline. For a more quantitative comparison, the measured centerline profile is fit to the analytical form

$$\phi_n(x) = A_n [\cosh(\beta_n x) - \cos(\beta_n x) - \eta_n (\sinh(\beta_n x) - \sin(\beta_n x))], \quad (4.9)$$

with

$$\eta_n = \frac{\cosh(\beta_n L) + \cos(\beta_n L)}{\sinh(\beta_n L) + \sin(\beta_n L)}. \quad (4.10)$$

To accommodate measurement uncertainties (exact clamp position and pixel origin), horizontal and vertical offsets (x_0, y_0) are included in the fit. Fit windows exclude outliers and regions with low sensitivity to avoid amplitude overestimation. Table 4.2 summarizes representative parameters from the fits in Fig. 4.4.

Mode	A_n (arb.)	L (px)	x_0 (px)	y_0 (arb.)
1	0.78	430.0	-17.0	0.06
2	0.27	423.0	-19.5	-0.01
3	0.33	388.8	10.1	0.03

Table 4.2 | Fit parameters of measured centerline profiles to Euler–Bernoulli mode shapes (Eqs. 4.9–4.10).

While the fit parameters for mode 1 and 2 are fairly consistent, the parameters for the third mode notably deviate from the others. The inferred lengths are consistent with the nominal ~ 400 px beam length. Overall, the analytical curves align well with the data outside excluded regions.

Discussion of deviations Deviations from the idealized model are attributable to the static deformations of the microfabricated beam. Regions where the beam is statically bent not only locally invalidate the model but also the small-signal assumption for increased out-of-plane displacements (due to reduced sensitivity). This combination of model deviation and increased measurement error is especially observable near the tip in the line profile of the first mode where the displacement amplitude is largest (Fig. 4.4).

4.6 Noise Analysis

A direct noise-floor curve (amplitude vs. drive at fixed frequency) would require strict pixel correspondence across repeated acquisitions, which is presently limited by uncorrected lateral drift. As a qualitative alternative, three measurements with different averaging: $N = 1$ vs. $N = 9$ vs. $N = 30$ frames per phase step (holding all other parameters fixed) were performed at $f = 158\text{kHz}$, where the beam is in its third mode. As in this mode two nodal lines are present, the decent of the amplitude values in these regions can be observed in from a profile line.

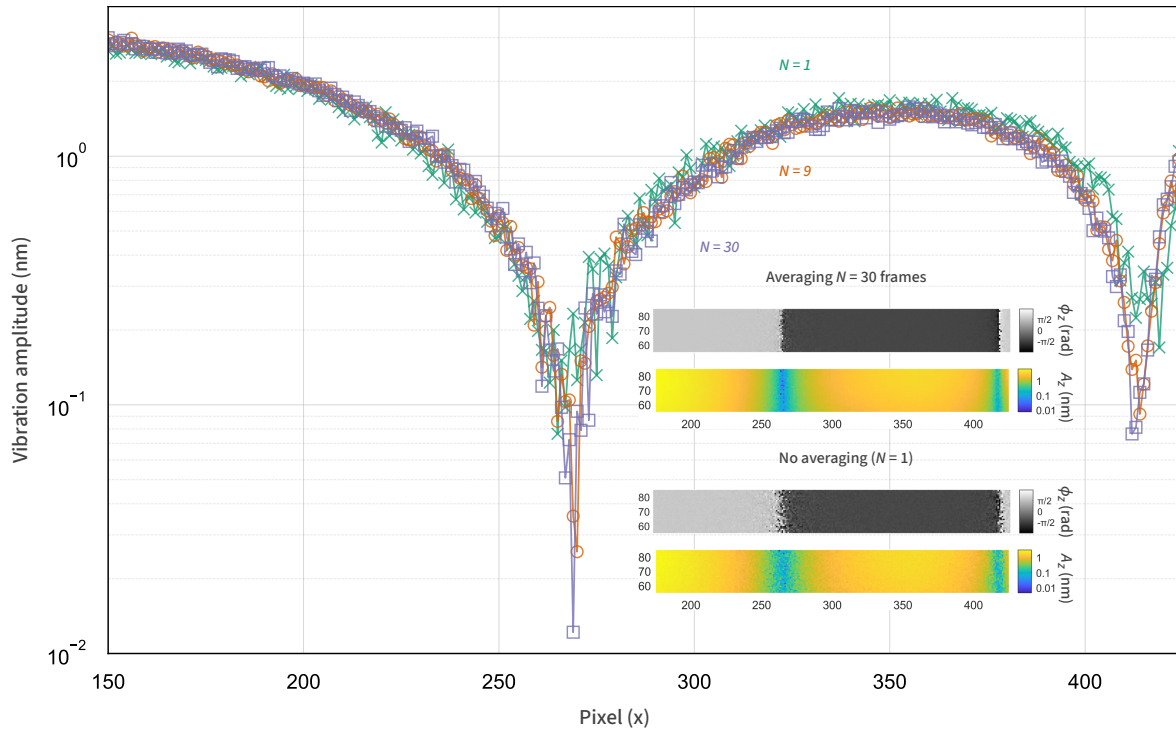


Figure 4.5 | Profile line (at $y = 70$) for different averaging numbers N of the beam at the third bending mode. Profiles from the averaged frame acquisition ($N = 9$ and $N = 30$) resolve the nodal transitions notably sharper than no averaging ($N = 1$). Inset amplitude and phase maps provide further qualitative insight of SNR improvement via frame averaging.

The profile lines along $y = 70$ (Fig. 4.5) show that the averaged acquisitions result in visibly cleaner nodal transitions with narrower and deeper spikes. While $N = 30$ does give slightly better results than $N = 9$, the improvement is not significant. Yet, theory predicts an SNR improvement factor of $\sqrt{30/9} \approx 1.8$, thus very likely the slow lateral drift eventually sets a bottleneck for the SNR improvement with averaging. The inset amplitude and phase maps in Fig. 4.5 give additional qualitative insight of the SNR improvement with respect to averaging. Using these results, an approximate value for the smallest detectable amplitude can be quantified to ~ 200 pm.

4.7 Summary

The experimental results demonstrate that the developed full-field interferometric system resolves the spatial mode shapes of MEMS cantilevers over a wide frequency range. Multiple resonances and their corresponding mode shapes were identified, and normalized frequency ratios and nodal structures exhibit good agreement with Euler–Bernoulli predictions.

Sensitivity analysis indicates that sub-nanometric amplitudes are detectable down to approx. 200 pm, with the effective noise floor limited by a combination of optical noise, camera characteristics, and environmental disturbances. Overall, these results validate the measurement approach for characterizing micro-acoustic MEMS devices and motivate incremental improvements, especially lateral stability to allow for longer averaging windows for further noise reduction.

In addition to the MEMS cantilever sample, the measurement system was applied to a variety of similarly dimensioned micro-acoustic samples yielding similar results in terms of sensitivity for frequencies up to 1 MHz.

5 Measurement System Software Development

The objective of this chapter is to describe the development of two software implementations that acquire and visualize interferometric data from the measurement system (Ch. 3): one implemented in MATLAB and one in Python. Both implementations use a shared JavaScript (React) GUI for a consistent user experience.

MATLAB's App Designer simplifies graphical UI creation, but native MATLAB figures and App Designer components are not well-suited for high-rate, real-time rendering. To achieve responsive interaction while keeping all scientific computation in MATLAB, a React-based GUI was embedded via the HTML UI Component, enabling bidirectional communication between JavaScript and MATLAB. For Python, the same React GUI was integrated through a FastAPI backend that serves the frontend and streams data/state in real time. This setup enables a like-for-like comparison of performance, usability, and flexibility across the two backends.

The GUI design is briefly described for completeness; the focus of this chapter is the backend design: stabilization-aware acquisition, low-latency data dispatch to the frontend, and real-time processing in MATLAB and Python.

5.1 GUI Design and Workflow

The GUI supports the end-to-end workflow (Fig. 3.5): device setting configuration, calibration sweep, operating-point selection, stabilization, acquisition (phase-stepping), and live plotting¹. A screenshot of the React-based interface is shown in Fig. 5.1. Panels include (i) device status and configuration (camera, signal generator), PID controls; (ii) calibration and acquisition controls and (iii), (iv), (v) (real-time) plots.

Data display is implemented with Plotly.js enabling interactive ROI selection, responsive pan/zoom and hover tooltips. Material-UI components provide a consistent, accessible layout. The frontend is event-driven: it subscribes to server-sent state updates (e.g., stabilization error, ROI means, PID control), and it publishes user actions (e.g., start/stop, parameter edits) back to the backend.

5.2 MATLAB Implementation

5.2.1 Single-thread execution model and implications

A single MATLAB session executes one callback at a time on the main thread. Although MATLAB can leverage parallel workers (e.g., `parfeval`, `backgroundPool`) for concurrent computation, worker

¹Live camera frame display has not yet been implemented

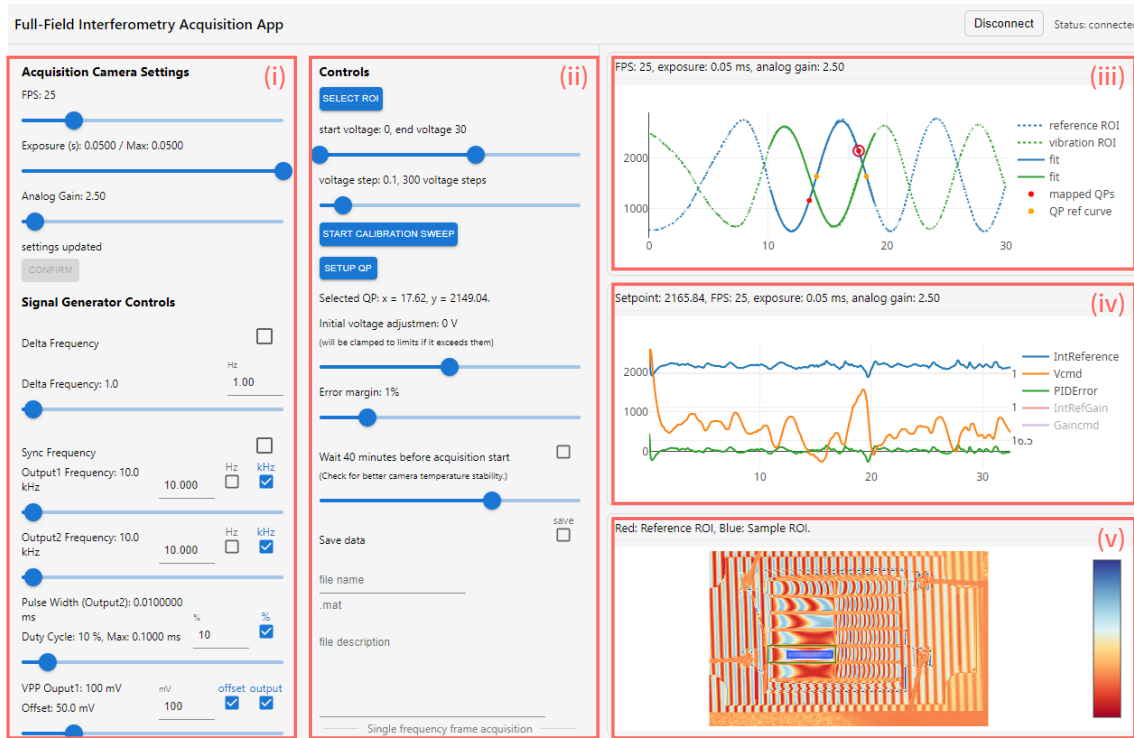


Figure 5.1 | Javascript React GUI. (i): Device configuration (camera and signal generator) as PID settings. (ii): Calibration controls and acquisition settings. (iii): Piezo control vs. ROIs plot window. (iv): real-time stabilization feedback plot. (v): Sample preview for interactive ROI selection.

threads cannot directly mutate main-thread state; data must be passed via futures with completion callbacks. Likewise, timer callbacks are queued and executed serially; `drawnow` (or event points like `pause`) flush the event queue at deterministic locations. While this single-threaded model (Fig. 5.2) ensures thread safety (eliminating race conditions and shared-state corruption), it severely limits concurrent task execution.

In this system, camera I/O callbacks, stabilization updates (PID), data buffering, and UI dispatch share a single event loop. As a result, conceptually independent tasks must run sequentially. Native App Designer plotting introduces additional overhead—particularly at frame rates in the tens of FPS. To maintain responsiveness, rendering is offloaded to the HTML UI Component (React + Plotly), while MATLAB focuses on numerical computation and lightweight data serialization. This separation of concerns enables a responsive MATLAB-based acquisition application.

5.2.2 High-level system architecture

The MATLAB backend follows a class-based architecture. Each physical instrument (camera, piezo stage, signal generator) is encapsulated in a dedicated controller class responsible for initialization, configuration, and low-level communication. A central `AcquisitionSystem` coordinates calibration, stabilization, acquisition, and processing. Stabilization is handled by `PIDController`; a `PlotController` rate-limits publishing of processed data to the GUI via the HTML UI Component.

The React frontend is embedded using MATLAB's HTML UI Component (`uihtml`) within a `uifigure`. On the JavaScript side, the application entry point defines `setup(htmlComponent)`. Events from the frontend use `sendEventToMATLAB(eventName, payload)`, which triggers the MATLAB-side `HTMLEventReceivedFcn`. Updates from MATLAB to JavaScript are performed by assigning JSON-serializable payloads to `uihtml.Data`, consumed by an `onData` handler. A practical detail: successive

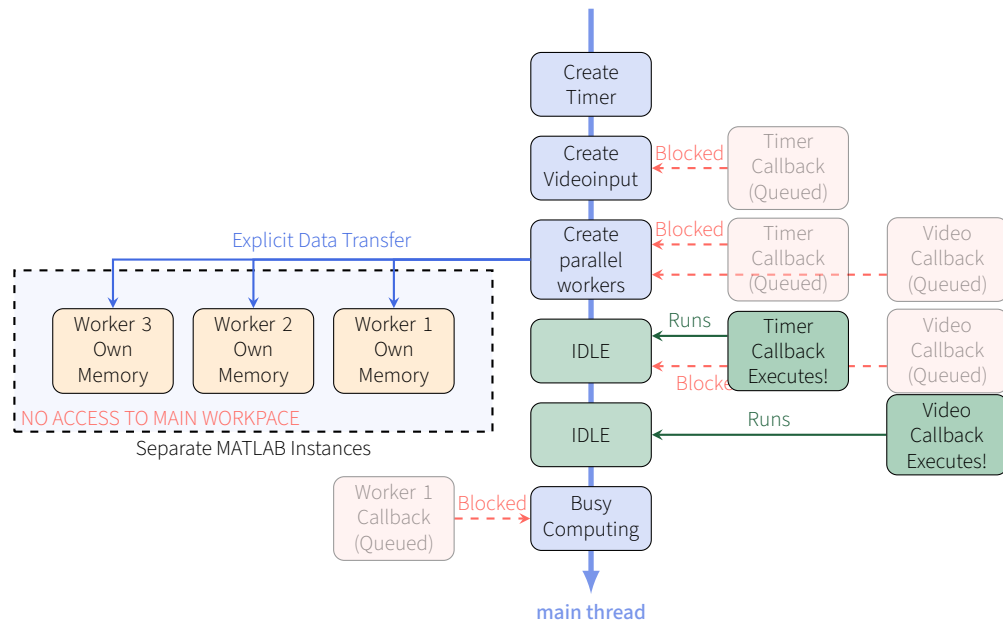


Figure 5.2 | MATLAB single thread execution model. Callbacks only execute when main thread is idle or forced using the **drawnow** command. As parallelized code execution such as **parfor** or **parfeval** act as separate MATLAB instances, they are primarily useful for concurrent computations of independent data.

assignments of identical data to `uihtml.Data` may not retrigger `onData`; including a monotonically increasing sequence number (e.g., `seq = seq + 1`) in each payload ensures delivery.

The App class owns the `uihtml` component and mediates interactions with `AcquisitionSystem`. Passing the `uihtml` handle to `PlotController` enables real-time visualization without coupling UI logic to acquisition or processing code.

Figure 5.3 illustrates module boundaries and message flow. The constructor of App (below) exemplifies this design.

App constructor

```
classdef App < handle
    properties
        UIFigure matlab.ui.Figure
        Html      matlab.ui.control.HTML
        % Physical Devices
        SigGen SigGenController
        Cam    CameraController
        Piezo  PiezoController
        % Acquisition / processing / plotting
        AQS    AcquisitionSystem
        Plt    PlotController
    end

    methods
        function app = App(htmlSource)
            app.UIFigure = uifigure('Name', 'Acquisition App');
            app.Html      = uihtml(app.UIFigure);
            app.Html.HTMLSource = htmlSource;
            app.Html.HTMLEventReceivedFcn = @(~,e) app.onHtmlEvent(e);

            % Build objects
            app.SigGen = SigGenController();
            app.Cam    = CameraController();
            app.Piezo  = PiezoController();
        end
    end
end
```

```

app.AQS    = AcquisitionSystem(app.SigGen, app.Cam, app.Piezo);
app.PlT    = PlotController(app.AQS, app.Html);

% Initial UI state
state = app.getInitialState();
state.seq = 0;           % monotonically increasing sequence
app.Html.Data = state;
end
end
end

```

5.2.3 Stabilization-aware acquisition and GUI dispatch

Stabilization-aware frame acquisition with real-time UI feedback requires the repeated execution of several tightly coupled tasks: (1) ingesting a new frame and its timestamp; (2) computing the reference ROI mean; (3) updating the PID controller and issuing a new piezo command; (4) managing indices and buffering acquired data; and (5) dispatching the latest results to the GUI. Although frame acquisition is the primary objective, neither per-frame PID control nor per-frame GUI updates are strictly required for correct operation.

Profiling revealed that GUI updates constitute the dominant performance bottleneck. To mitigate this, `PlotController` rate-limits GUI dispatch using a MATLAB timer that fires at a fixed interval (e.g.,

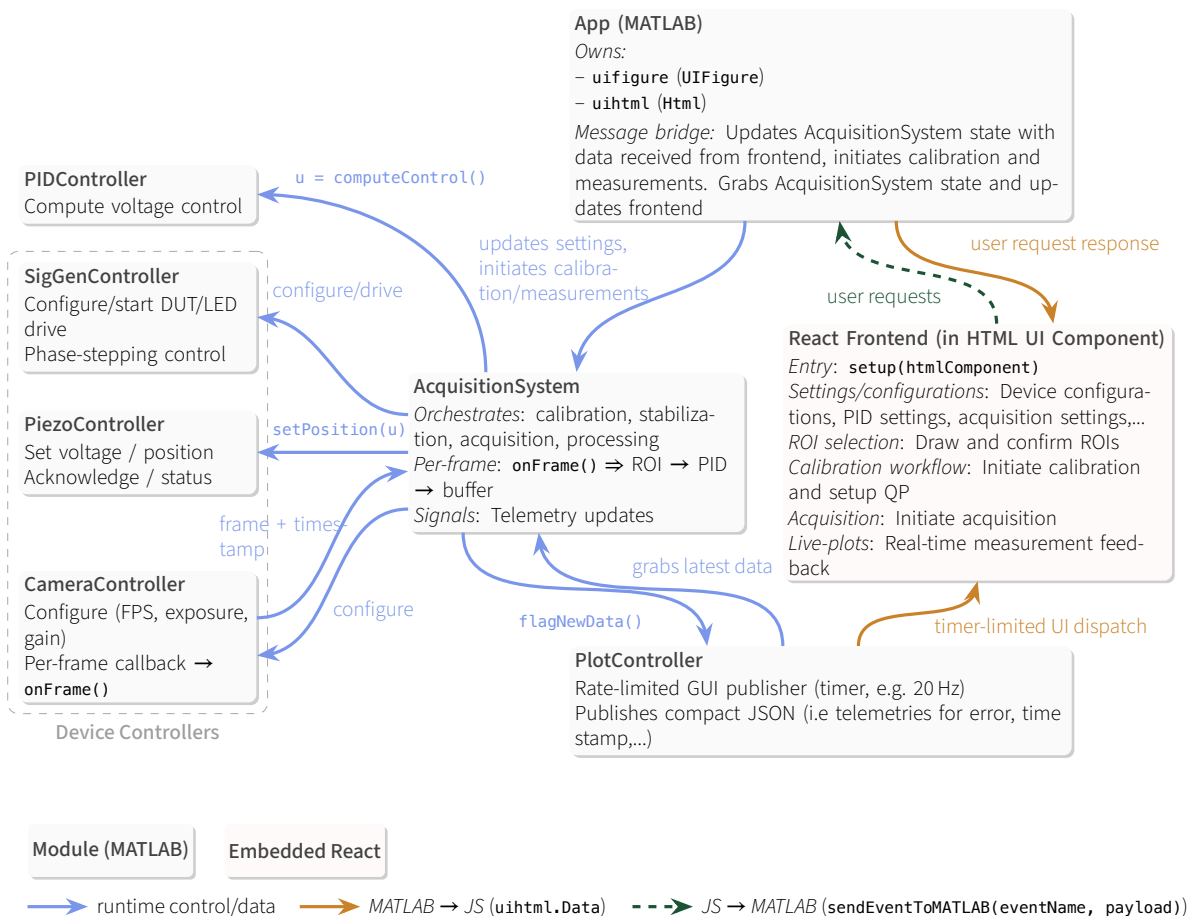


Figure 5.3 | Software architecture diagram of the MATLAB code base. The `AcquisitionSystem` acts as a central orchestrator of acquisition relevant components. It is bridged with the JS (React) frontend via the `App` module. The `PlotController` has access to the `AcquisitionSystem` and dispatches data directly to the frontend.

20 Hz). Each tick checks a “new data available” flag set by the acquisition loop (via a MATLAB `notify` event). If set, the latest summaries are sent to the GUI and the flag is cleared. This prevents GUI backpressure from stalling acquisition.

In principle, the PID feedback loop could be rate-limited using the same mechanism. However, MATLAB timers restrict callback execution to a maximum frequency of approximately 20 Hz, whereas the PID loop only becomes a bottleneck at frame rates above roughly 50 FPS. Since limiting the PID update rate yields little performance benefit while negatively affecting stabilization quality, PID feedback is therefore executed on every acquired frame.

As discussed in Sec. 5.2.1, tasks (1)–(5) must be executed sequentially due to MATLAB’s single-threaded event loop. This constraint simplifies the acquisition architecture: a single callback is triggered for each incoming frame and executes the full acquisition pipeline in order. With this design, the current MATLAB backend sustains approximately 25 FPS. In the following this implementation logic is outlined.

Measurement initiation A measurement is initiated via the `startMeasurement()` function, which pre-allocates buffers, resets the signal generator, piezo actuator, and PID state, and signals the `PlotController` to start its rate-limited timer. The acquisition-mode-specific callback is then registered with the camera and invoked for every incoming frame:

Start measurement: setup and bind callback (owned by `AcquisitionSystem`)

```
function startMeasurement(obj, varargin)
    % Initialize
    obj.isMeasuring = true;
    obj.resetBuffers();

    % Start rate-limited PlotController timer (e.g., 20 Hz)
    obj.Plot.start();

    % Configure hardware
    obj.Camera.configureForAcquisition(obj.CameraSettings);
    obj.Piezo.setVoltage(obj.InitialVoltage); % avoid ambiguous feedback phase
    obj.SigGen.configureForAcquisition(obj.SigGenSettings);

    % Initialize PID time base
    obj.PID.reset();

    % Ensure init callbacks execute before frames arrive
    drawnow;

    % Arm camera with per-frame callback
    obj.Camera.startAcquisition(@obj.onFrame);
end
```

Core per-frame callback The per-frame `onFrame` callback, implemented as part of the `AcquisitionSystem`, has access to all active devices and shared state. After processing and buffering the current frame, it signals the `PlotController` that new data are available via `flagNewData()`. The `PlotController` publishes data to the GUI in batches. Each dispatch records the index of the most recently transmitted sample; when the timer fires again, all data newer than this index are sent to the frontend, ensuring efficient and lossless updates even when GUI dispatch is rate-limited.

Per-frame callback (stabilization + buffering)

```
function onFrame(obj, varargin)
    if ~obj.isMeasuring, return; end
    i = obj.frameIndex;

    % 1) Frame + timestamp
```

```

[frame, ts] = obj.Camera.getLatestFrame();

% 2) ROI mean for stabilization
muRef = obj.getMuRef(frame);

% 3) PID update and piezo command
[u, err] = obj.PID.computeControl(muRef);
obj.Piezo.setPosition(u);

% 4) Telemetry record
obj.ts(i)    = ts;
obj.muRef(i) = muRef;
obj.err(i)   = err;

% Accept only near-setpoint frames into phase buffer
if abs(err) < obj.PID.acceptance
    k = obj.phaseIndex;
    obj.phaseStack(:, :, k) = frame; % preallocated [Y X M]
    obj.phaseIndex = mod(k, obj.numPhases) + 1;

    % Optionally: stop when full stacks acquired
    % if obj.phaseIndex == 1, obj.onPhaseBlockComplete(); end
end

% 5) Signal GUI updater (rate-limited timer will pick this up)
obj.flagNewData();

obj.frameIndex = i + 1;
end

```

5.3 Python Implementation

5.3.1 High-level system architecture

At a high level, the Python implementation mirrors the MATLAB structure (controller classes per instrument plus a central orchestrator). A FastAPI backend serves the React GUI and handles back-end processing. User interactions arrive via standard HTTP endpoints; live data stream to the frontend over WebSockets for low-latency visualization. Here, the discussion is limited to the stabilized acquisition using Python’s multi-threading.

5.3.2 Multi-threaded stabilization-aware acquisition and GUI dispatch

Tasks (1)–(5) from Sec. 5.2.3 are predominantly I/O-bound (frame ingest, device commands, GUI I/O). Python threads (threading) are therefore suitable despite the Global Interpreter Lock (GIL), because threads release the GIL when blocking on I/O. Multiprocessing can provide true parallelism but increases complexity due to inter-process communication (IPC). To balance simplicity and responsiveness, the implementation uses threads plus bounded queues with a “skip-full” policy to prevent backpressure.

Separate threads handle camera acquisition, piezo control, and GUI updates. `queue.Queue` provides thread-safe communication; a producer (camera callback) uses non-blocking `put_nowait` into size-1 queues (skips on full queues), while consumers (piezo and GUI update callback) block on `get()`. An acquisition rate of ~85 FPS was achieved with this architecture (details depend on hardware and drivers), and is detailed in the following.

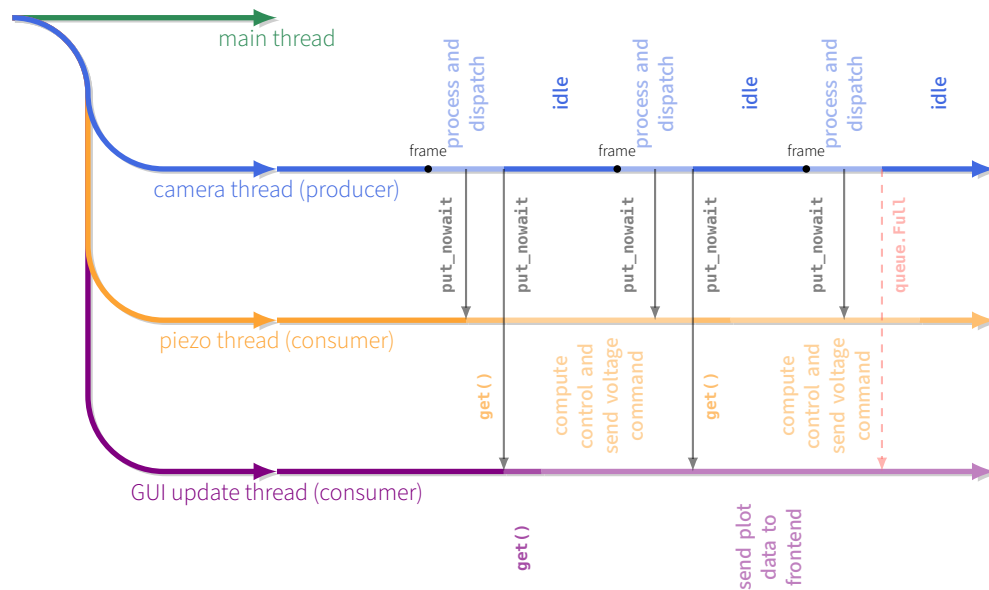


Figure 5.4 | Python multi-threaded acquisition (timeline). The camera thread dispatches to bounded queues (`size=1`, skip-full via `put_nowait`), while piezo and plot threads consume via blocking `get()`.

Measurement Initiation The `start_measurement()` function initializes a measurement by preallocating buffers and resetting the signal generator, piezo actuator, and PID controller state. Since the callbacks executed by the camera and piezo threads depend on the acquisition mode, these threads are started with acquisition-specific callbacks defined within the `AcquisitionSystem` class. As a result, data queues do not need to be explicitly passed into these callbacks.

In contrast, the GUI update thread is independent of the acquisition mode. It receives a list of variables to be visualized and a data queue from which it consumes packets to be forwarded to the frontend.

Start measurement: Initiate threads and bind callbacks (owned by `AcquisitionController`)

```
def start_measurement(self) -> None:
    # Bounded queues (skip-full policy implemented by non-blocking put)
    self._piezo_q = queue.Queue(maxsize=1)
    self._data_q = queue.Queue(maxsize=1)

    # Latest control value shared with producer
    self._acq_latest_lock = threading.Lock()
    self._latest_v_cmd = self._init_v

    # Configure devices and PID
    self._configure_camera()
    self._configure_siggen()
    self._pid_controller.initialize()

    # Allocate storage
    self._reset_buffers()

    # Start streaming + async
    self.camera.start()
    self.camera.start_async(
        on_sample=self._on_frame_acquisition
    )

    # Set initial voltage
    self.piezo.start_async(
        async_target=self._acquisition_piezo_cb,
```

```

    initial_voltage=self._init_v
)

# Start plotting thread
self.plot_controller.start_async(
    data_q=self._data_q,
    plot_options=self._plt_opts
)

self._running = True

```

Camera thread The `start_async` method of the `CameraController` continuously retrieves the latest frame from the camera buffer and forwards it to the registered callback. Frame retrieval blocks until new data are available, ensuring efficient synchronization with the camera driver.

Camera thread initiation (owned by `CameraController`)

```

def start_async(self, on_sample: Callable[[np.ndarray, int], None]) -> None:
    self._stop_evt = threading.Event()
    def _worker():
        while not self._stop_evt.is_set():
            frame, t_ns = self.get_fast_frame() # blocks until frame arrives
            on_sample(frame, t_ns)
    t = threading.Thread(target=_worker, name="CameraAsync", daemon=True)
    t.start()
    self._thread = t

```

Piezo thread The piezo control thread retrieves the most recent stabilization error, computes the corresponding control command, and applies it to the actuator. A short sleep interval provides settling time and reduces CPU utilization. The GUI update thread follows a similar structure, with a longer sleep interval chosen to enforce an update rate of approximately 20 Hz.

Piezo thread initiation (owned by `PiezoController`)

```

def start_async(self, *, async_target: Callable[[], float],
    initial_voltage: float = 0.0) -> None:
    self.set_voltage(initial_voltage)
    self._stop_evt = threading.Event()
    def _worker():
        while not self._stop_evt.is_set():
            v_cmd = async_target() # compute next command
            self.set_voltage(v_cmd)
            time.sleep(0.01) # settle + yield
    t = threading.Thread(target=_worker, name="PiezoAsync", daemon=True)
    t.start()
    self._thread = t

```

Camera frame callback (producer) The camera frame callback processes incoming frames, updates telemetry buffers, enqueues the latest stabilization error for the piezo thread, and pushes compact data batches to the GUI queue. The bounded queues ensure that consumers always operate on the most recent state, regardless of temporary rate mismatches between producer and consumers.

This callback largely determines the maximum achievable acquisition rate. Piezo updates and GUI dispatch are both driven by newly processed frames, and their callbacks are predominantly I/O-bound. Together with the overhead of retrieving frames from the camera buffer, this function therefore defines the performance ceiling of the acquisition loop.

Numeric-only callback called from camera thread (owned by `AcquisitionController`)

```

def _on_frame_acquisition(self, frame: np.ndarray, t_ns: int) -> None:
    i = int(self._i)

```

```

# Process frame: compute mu_ref, update phase buffer, etc.
mu_ref = self._process_frame(frame)
ts_s = t_ns * 1e-9

with self._acq_latest_lock:
    v_cmd_prev = self._latest_v_cmd
    e = mu_ref - self._pid_setpoint

# Store telemetry
self._t_s[i] = ts_s
self._v_cmd[i] = v_cmd_prev
self._mu_ref[i] = mu_ref
self._pid_e[i] = e

# Latest error for piezo thread (skip-full policy)
if not self._piezo_q.full():
    try:
        self._piezo_q.put_nowait({"e": e})
    except queue.Full:
        pass

# Compact batch for GUI (skip-full policy)
start = self._last_put_idx
end = i + 1
if end > start and not self._data_q.full():
    pkt = {
        "t": self._t_s[start:end].tolist(),
        "v_cmd": self._v_cmd[start:end].tolist(),
        "mu_ref": self._mu_ref[start:end].tolist(),
        "pid_e": self._pid_e[start:end].tolist(),
        "start": start,
        "end": end,
    }
    try:
        self._data_q.put_nowait(pkt)
        self._last_put_idx = end
    except queue.Full:
        pass

self._i = end

```

Piezo callback (consumer) The piezo callback consumes the most recent stabilization error produced by the camera thread, computes the corresponding control voltage using the PID controller, and returns it to the PiezoController for application to the actuator.

Piezo thread callback: compute control command (owned by AcquisitionController)

```

def _acquisition_piezo_cb(self) -> float:
    latest = self._piezo_q.get() # blocks until available
    e = float(latest["e"])
    u = self._pid_controller.step(e)
    with self._acq_latest_lock:
        self._latest_v_cmd = u
    return u

```

5.4 Comparison and Summary

Both the MATLAB and Python implementations deliver on the initial objectives. A shared JavaScript (React) frontend decouples visualization from acquisition and processing in both backends, keeping them focused on numeric work and lightweight data serialization. MATLAB achieves this via the ui-

html component embedded in a `uifigure`, while Python uses a FastAPI backend that serves the React app and streams state via WebSockets. The high-level modularization is similar across both versions (device controllers, central orchestrator, plotting/publishing), but the stabilization-aware acquisition workflows differ:

MATLAB A single-threaded control path executes stabilization, buffering, and rate-limited GUI dispatch sequentially (dispatch throttled by a timer).

Python The threading module separates concerns across camera acquisition, piezo control, and GUI publishing, communicating via bounded queues (skip-full producer semantics).

On the reference hardware, sustained acquisition rates of approximately ~ 25 FPS (MATLAB) and ~ 85 FPS (Python) were achieved. The performance gap primarily reflects the concurrency model and message/plotting pipeline: MATLAB's single-threaded event loop and the cost of frequent UI bridge updates favor lower dispatch rates; Python's threaded design and WebSocket streaming allow higher throughput at the cost of additional concurrency and server complexity.² The main differences between the versions are summarized in Tab. 5.1.

	MATLAB	Python
Architecture	Single-session, single-threaded event loop; HTML UI Component bridge	FastAPI backend; threaded workers (camera / piezo / GUI); WebSocket to React
Concurrency model	Deterministic sequencing; optional <code>timer</code> for rate limiting	Natural separation of concerns; shared-memory threads with <code>queue.Queue</code> (bounded, skip-full); blocking consumers
UI bridge	<code>uihtml.Data</code> (JSON) + <code>sendEventToMATLAB</code> ; requires sequence counter to retrigger handlers on identical payloads	JSON over WebSocket; server-sent rate control and message coalescing
Throughput (ref. hardware)	~ 25 FPS sustained with decimated UI payloads	~ 85 FPS sustained with bounded queues and compact packets
Strengths	Simple, predictable control flow; tight integration with device SDKs and MATLAB numerics; minimal moving parts	High throughput and low-latency streaming; flexible I/O; clean isolation of acquisition vs. publishing; easier background processing
Limitations	UI dispatch can dominate runtime; single-threaded loop couples I/O and GUI; limited visibility into JS errors unless instrumented	More moving parts (server, threads, queues); concurrency adds failure modes; WebSocket/serialization and shutdown semantics need care
Error handling	Robust main-loop safety; JS errors not surfaced automatically without explicit handlers; watchdogs needed for device timeouts	Explicit try/catch per thread; queue backpressure, and graceful shutdown required
Best fit	Rapid development where determinism and simplicity outweigh max throughput; tight MATLAB-based analysis pipelines	Higher acquisition rates or richer streaming; modular services; future scale-out

Table 5.1 | Comparison of MATLAB and Python implementations. Qualitative differences reflect the concurrency model and UI bridge.

²Measured rates are device-, driver-, and PC-dependent; results reflect the configuration described in Ch. 3.

6 Conclusions and Future Work

6.1 Summary of Achievements

With respect to the objectives formulated at the outset of this project (Secs. 1.1 and 1.2), the results presented in this thesis can be considered a success. The developed Michelson-type full-field interferometric setup performs stabilized measurements from which vibration maps are retrieved via a small-signal linearization. The software-driven stabilization—ensuring a constant per-pixel sensitivity mapping throughout the measurement—is preceded by a rigorous calibration procedure that finds the systems optimal stabilization point referred to as quadrature point at which sensitivity is maximized. Stroboscopic illumination allows for capturing an arbitrary number of frames at constant vibration phase enabling (shot) noise reduction by averaging. In addition, the stabilization feedback error allows for low-quality frame rejection, further enhancing robustness.

Vibration measurements of in-house MEMS samples were performed at frequencies of up to 1 Mhz. Combined with the developed signal processing pipeline, these measurements produce physically consistent full-field amplitude and phase maps. A sensitivity analysis further shows that the system fulfills the target of sub-nanometer displacement sensitivity, with a practical noise floor of approximately 200 pm.

The custom control and acquisition software for the measurement system was realized in both MATLAB and Python leveraging different architectures each with their individual strengths. Both backends are complemented by a modern user-friendly (JS-React-based) graphical user interface allowing for interactive calibration and real-time measurement feedback.

Finally, this work shows that a custom built full-field vibrometer is indeed a cost-effective solution by combination of accessible hardware: Beyond the optical components required for a Michelson-type interferometer (including magnification), only a signal generator and piezo voltage controller are required—instruments not specific to this application alone and usually equip most modern laboratories.

6.2 Limitations and Future Directions

While the main objectives have been achieved, the developed system still offers considerable potential for improvement and extension. Three main areas for future work have been identified.

(i) Lateral stability As discussed in the introduction, state-of-the-art full-field vibrometry setups can reach displacement sensitivities on the order of $\lesssim 30$ pm. In the present setup, lateral drift and creep currently limit the extent to which sensitivity can be improved through frame averaging. This can be addressed via mechanical and/or software-based solutions. Moreover, brighter illumination—for example through higher-current pulsed operation enabled by an external LED driver—would increase the detected photon number and thereby reduce the relative impact of shot noise.

(ii) Low-frequency measurements The measurement of very low frequency vibrations has not yet been addressed. In principle, stroboscopic illumination can also be applied at low vibration frequencies. In practice, however, the present stabilization pipeline is closely coupled to the camera-based feedback signal. Since conventional PID-based stabilization becomes increasingly ineffective as the relevant feedback dynamics approach DC, stable operation is compromised in this regime. Consequently, dedicated strategies for low-frequency measurements need to be explored, either by modifying the measurement modality or by redesigning the stabilization procedure.

(iii) Relaxing the sample-flatness requirement With the current calibration procedure, significant deviations between the local operating point and the local quadrature point may occur if the sample surface is not sufficiently flat. In such regions, the small-signal linearization becomes less accurate and introduces larger reconstruction errors. One possible extension would be an operating-point sweep that allows slightly curved surfaces to be measured more reliably. For larger surface-height variations—in particular when the OPD variation exceeds approximately one quarter of the interference period—it may become necessary to employ multiple phase-shifted reference regions.

6.3 Summary of Contributions and Concluding Remarks

During a six-month tenure at Qualcomm RF360, the author developed the complete control software and data-processing pipeline of the measurement system from the ground up. This includes device communication, calibration routines, stabilization logic, measurement automation, and vibration-data evaluation. All illustrations presented in this thesis were created by the author and are original works subject to copyright.

Bibliography

- [1] Stephen D. Senturia. *Microsystem Design*. Boston: Kluwer Academic Publishers, 2001.
- [2] Nadim Maluf and Kirt Williams. *An Introduction to Microelectromechanical Systems Engineering*. 2nd ed. Boston: Artech House, 2004.
- [3] Chang-Hoon Liu. "CMOS MEMS Fabrication Technologies and Devices". In: *Micromachines* 7.1 (2016), p. 14. doi: **10.3390/mi7010014**.
- [4] M. Lemkin and B. E. Boser. "A surface micromachined silicon accelerometer with on-chip detection circuitry". In: *Sensors and Actuators A: Physical* 45.1 (1994), pp. 7–16. doi: **10.1016/0924-4247(94)00815-9**.
- [5] J. Bustos et al. "Technology for the high-volume manufacturing of integrated surface-micromachined accelerometer products". In: *Microelectronics Journal* 29.9 (1998), pp. 579–586. doi: **10.1016/S0026-2692(98)00021-4**.
- [6] Ahmad Safari and E. Koray Akdoğan, eds. *Piezoelectric and Acoustic Materials for Transducer Applications*. New York, NY: Springer Science & Business Media, 2008. ISBN: 978-0-387-76540-2. doi: **10.1007/978-0-387-76540-2**.
- [7] Reza Abdolvand et al. "Micromachined Resonators: A Review". In: *Micromachines* 7.9 (2016), p. 160. doi: **10.3390/mi7090160**.
- [8] Tianren Feng et al. "Concepts and Key Technologies of Microelectromechanical Systems Resonators". In: *Micromachines* 13.12 (2022), p. 2195. doi: **10.3390/mi13122195**.
- [9] Kenji Uchino. *Advanced Piezoelectric Materials: Science and Technology*. 2nd. Cambridge, MA: Woodhead Publishing, 2017. ISBN: 9780081012543.
- [10] Yanmei Xue et al. "Acoustic wave devices based on piezoelectric/ferroelectric thin films for high-frequency communication systems and sensing applications". In: *Ceramics International* 50.23, Part C (2024). The 13th Asian Meeting on Ferroelectrics jointly with the 13th Asian Meeting on Electroceramics, pp. 52051–52058. ISSN: 0272-8842. doi: **<https://doi.org/10.1016/j.ceramint.2024.03.212>**. URL: **<https://www.sciencedirect.com/science/article/pii/S0272884224011647>**.
- [11] Vyacheslav Aranchuk et al. "Multi-beam heterodyne laser Doppler vibrometer based on a line-scan CMOS digital camera". In: *Appl. Opt.* 61.20 (July 2022), pp. 5876–5883. doi: **10.1364/AO.461368**. URL: **<https://opg.optica.org/ao/abstract.cfm?URI=ao-61-20-5876>**.
- [12] Cameron Sepehr Marashi, Philip Bradford, and Kara Peters. "Laser Doppler vibrometry measurements of acoustic attenuation in optical fiber waveguides". In: *Appl. Opt.* 62.16 (June 2023), E119–E124. doi: **10.1364/AO.483827**. URL: **<https://opg.optica.org/ao/abstract.cfm?URI=ao-62-16-E119>**.
- [13] Kimmo Kokkonen and Matti Kaivola. "Scanning heterodyne laser interferometer for phase-sensitive absolute-amplitude measurements of surface vibrations". In: *Applied Physics Letters* 92.6 (Feb. 2008), p. 063502. doi: **10.1063/1.2840183**. URL: **<https://doi.org>**.
- [14] Erlend Leirset, Helge E. Engan, and Astrid Aksnes. "Heterodyne interferometer for absolute amplitude vibration measurements with femtometer sensitivity". In: *Optics Express* 21.17 (Aug. 2013), pp. 19900–19913. doi: **10.1364/OE.21.019900**. URL: **<https://opg.optica.org>**.
- [15] Lauri Lipiäinen, Kimmo Kokkonen, and Matti Kaivola. "Homodyne full-field interferometer for measuring dynamic surface phenomena in microstructures". In: *Optics and Lasers in Engineering* 88 (2017), pp. 178–183. ISSN: 0143-8166. doi: **<https://doi.org/10.1016/j.optlaseng.2016.08.004>**. URL: **<https://www.sciencedirect.com/science/article/pii/S0143816616301646>**.
- [16] Lauri Lipiäinen, Kimmo Kokkonen, and Matti Kaivola. "Stabilized Stroboscopic Full-Field Interferometer for Characterization of Subnanometer Surface Vibrations". In: *Journal of Microelectromechanical Systems* 24.5 (2015), pp. 1642–1646. doi: **10.1109/JMEMS.2015.2428433**.
- [17] Lauri Lipiäinen. "Optical imaging of surface dynamics in microstructures". English. PhD thesis. Finland: Aalto University, 2016. ISBN: 978-952-60-7051-3.
- [18] *VibroFlex Connect VFX-F-110*. Technical specifications are subject to change without notice. Polytec GmbH. Jan. 2026.

- [19] “Optical Profiling Techniques for MEMS Measurement”. In: *Optical Inspection of Microsystems*. CRC Press, Taylor & Francis Group, June 2019. Chap. 7, pp. 177–207. ISBN: 9780429186738. doi: <https://doi.org/10.1201/9780429186738>.
- [20] Kurt Meyberg and Peter Vachenauer. *Höhere Mathematik 2: Differentialgleichungen, Funktionentheorie, Fourier-Analysis, Variationsrechnung*. 4th ed. Springer-Lehrbuch. Springer Berlin Heidelberg, 2001. ISBN: 978-3-540-41851-1.
- [21] Athanasios Papoulis and S. Unnikrishna Pillai. *Probability, Random Variables, and Stochastic Processes*. 4th. Boston: McGraw-Hill, 2002. ISBN: 978-0073660110.
- [22] Eugene Hecht. *Optics*. 5th. Boston: Pearson Education, 2017. ISBN: 978-0133977226.
- [23] P. Hariharan. *Basics of Interferometry*. 2nd. Boston: Academic Press, 2007. ISBN: 9780123735898.
- [24] P. Hariharan. *Optical Interferometry*. 2nd ed. Amsterdam: Academic Press, 2007.
- [25] Joseph W. Goodman. *Statistical Optics*. 2nd ed. Hoboken, NJ: Wiley, 2015.
- [26] Wolfgang Osten. *Optical Inspection of Microsystems*. Optical Science and Engineering. CRC Press, Taylor & Francis Group, June 2019. ISBN: 9780429186738. doi: <https://doi.org/10.1201/9780429186738>.
- [27] R. L. Powell and K. A. Stetson. “Interferometric vibration analysis by wavefront reconstruction”. In: *Journal of the Optical Society of America* 55.12 (1965), pp. 1593–1598.
- [28] “Optical Profiling Techniques for MEMS Measurement”. In: *Optical Inspection of Microsystems*. CRC Press, Taylor & Francis Group, June 2019. Chap. 11, pp. 297–347. ISBN: 9780429186738. doi: <https://doi.org/10.1201/9780429186738>.
- [29] Alan V. Oppenheim and Ronald W. Schaffer. *Discrete-Time Signal Processing*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2010.
- [30] James R. Janesick. *Scientific Charge-Coupled Devices*. Bellingham, WA: SPIE Press, 2001.
- [31] *High Brightness LED, Ø 5 mm Untinted Non-Diffused Package*. VLCS5830. Rev. 1.3. Vishay Semiconductors. Apr. 2013.