https://doi.org/10.1038/s41596-023-00924-6

Protocol

Check for updates

Hyperspectral confocal imaging for high-throughput readout and analysis of bio-integrated microlasers

In the format provided by the authors and unedited

Hyperspectral confocal imaging for high-throughput readout and analysis of bio-integrated microlasers

Supporting Information

Supplementary Note 1: Design of the free-space confocal microscope

Here, we describe the construction of the different parts of a modular homebuilt confocal microscope. The confocal scan head that raster scans the pump laser over the sample can be attached to either a commercial or a homebuilt microscope. The microscope body itself will likely impose the strongest geometric constraints on the positioning of the optics; therefore, this part should be built or positioned first. In our system, we use a simple homebuilt inverted microscope consisting of a motorized stage (STAGE; Figure S1a) and basic transmission imaging capabilities (KM3, LP1, L3, CAM1). Our optical elements are designed as a cage system, predominantly using a 30mm cage system size. This provides alignment of the optical elements in the off-axis (x-y) direction, with integrated translation mounts for any components requiring additional fine-alignment. A 90:10 beamsplitter (FC2) is situated behind the objective turret to reflect 90% of the light into the confocal scanning path (seen as branching off to the left in Figure S1a). The remaining 10% are transmitted downward and then reflected forward by a kinematic mirror for alignment (KM3), pass through a long-pass filter to block the confocal scanning laser (LP1), and are focused with the brightfield tube lens (L3) onto a USB camera (CAM1).

The remainder of the optical elements are part of the confocal excitation and detection. We first build the main confocal excitation path (EXC1; Figure S1b). For excitation of laser particles (LPs), we recommend using lasers with repetition rates close to or in the MHz regime and sub-ns pulse durations. They should further provide pulse energies ca. 100-fold above the lasing threshold pulse energy of the LPs used to account for all potential losses (e.g., from overfilling the objective back aperture, imaging through scattering media, etc.) and still allow operation of the LPs well above threshold. The beam from the excitation laser first passes through a fluorescent filter cube (FC1) that houses a corner mirror insert, allowing to switch to additional excitation lines. Two kinematic mirrors (KM1, KM2) allow alignment of the main excitation line, and additional lasers (e.g., EXC2) can later be aligned to the main path with an additional set of kinematic mirrors. A set of filter wheels (FW) holds various neutral density filters (ND0-ND4 in the first filter wheel, ND0.0-ND0.6 in the second filter wheel) to adjust the laser power. The filter wheel used here cannot be housed inside the cage system; therefore, the cage rods are taken out at the position of the filter wheels. Next, the excitation laser beam is transmitted through a filter cube (FC3) housing the dichroic filter, which reflects the de-scanned signal from the sample.

The excitation beam then enters a typical confocal scanning configuration consisting of the 2D galvo scanner (GM; x-y scanning), a scan lens (L1; Figure S1a) attached to the galvo mirror mount, and the confocal tube lens (L2). As the scan lens expands the beam, a 60 mm cage system is used to house the optics in this part of the system. This cage system is connected with a cage system adapter to the 90:10 beamsplitter (FC2), where the confocal excitation is combined with the brightfield light path before the back aperture of the objective. The objective used for confocal scanning is mounted on a piezoelectric objective scanner (OS; z-scanning) sitting on a turret for more flexibility in magnification.

After exciting the LPs with the excitation laser, the lasing signal originating from the sample is collected by the same objective and follows the confocal path back until reaching the dichroic mirror in filter cube FC3. Here, the wavelength range of interest is reflected into the detection arm to couple the signal into the optical fiber (FIB) connected to the spectroscopy module (Figure S2a). The coupling can be aligned laterally using another set of kinematic mirrors (KM4, KM5) and the x-y-translation mount of the fiber. The coupling lens (L4) is translatable along z to be focused onto the end facet of the optical fiber such that the fiber core serves as a pinhole that rejects out-of-focus light for improved optical sectioning. This is the case when the fiber core has a diameter matching that of the airy disk (for our combination of relay lenses, this corresponds to a fiber with a 10 µm diameter core). The fiber is mounted on an SMA adapter plate to allow fast switching between different fibers; we have found that it is useful to additionally have access to a fiber with 105 µm diameter core which provides an ‘open pinhole’ configuration with higher signal intensity but reduced optical sectioning (Figure S14). Increasing the NA or the core size of the fiber further is not recommended because this will increase losses when coupling the signal into the spectrometer.

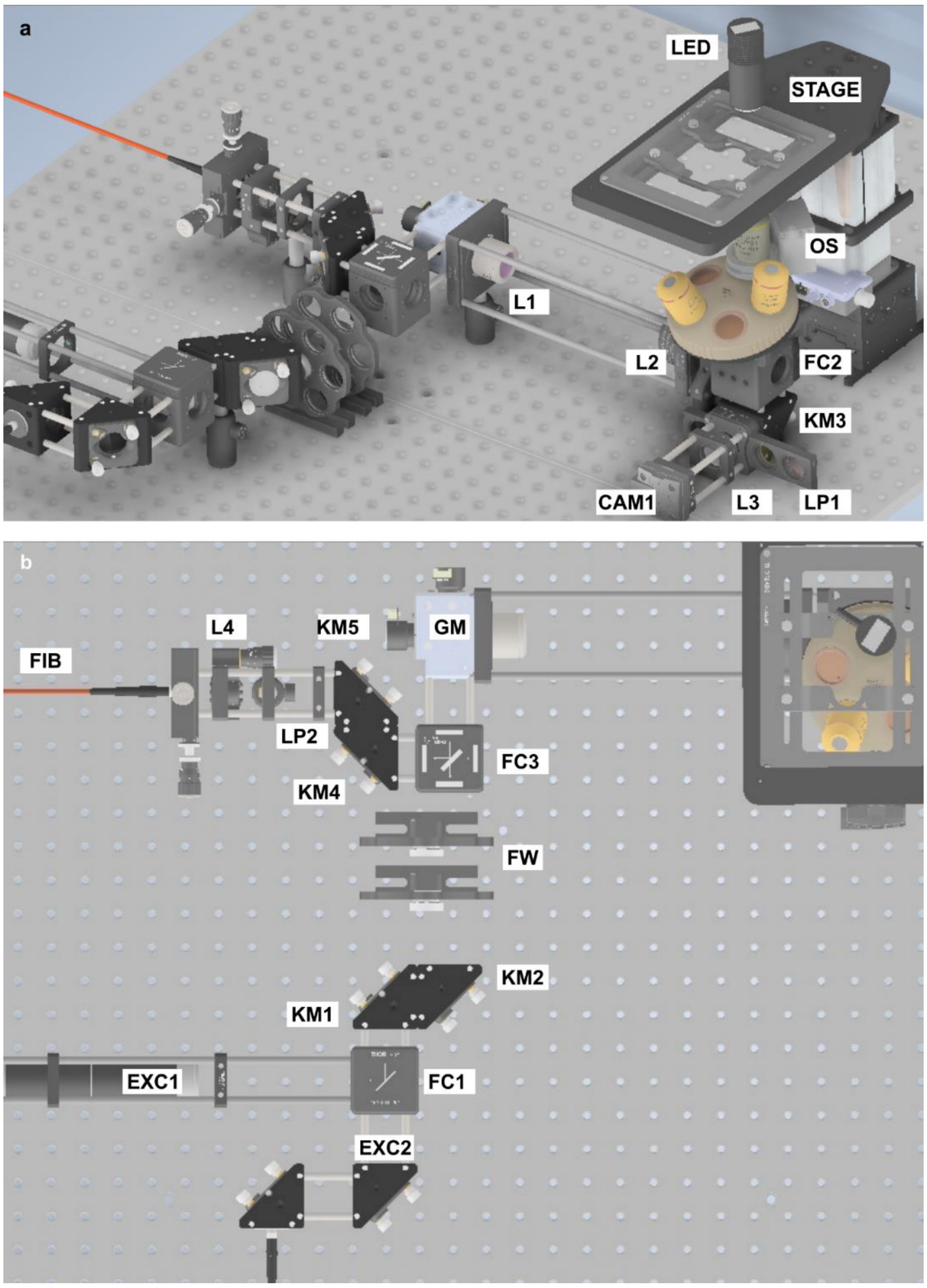


Figure S1 Annotated CAD rendering of the homebuilt confocal microscope. a, Side-view, and **b**, top-view.

The other end of the fiber is mounted in a small relay system forming an image of the fiber core in the optical plane of the spectrometer entrance slit. This allows closing the slit to improve spectral resolution without physically obstructing or even touching the slit with the fiber. The relay lenses can be chosen such that the signal emerging from the 0.1 NA optical fiber is matched to the NA of the parabolic mirrors inside the spectrometer (0.077) to prevent losses and aberrations. Therefore, the signal from the fiber is collimated with a *f* = 40 mm achromatic doublet and magnified onto the slit plane with a *f* = 50 mm achromatic doublet, which is mounted onto a fine z-translation mount to allow precise focusing. The SMA fiber adapter is fixed with an x-y translation mount for lateral alignment. Between the lenses, we have inserted another fluorescent cube (FC4) that is usually empty, but dichroic or dielectric mirrors can be inserted if desired, e.g., to couple light into the fiber during alignment, or to leave flexibility for an additional detection path.

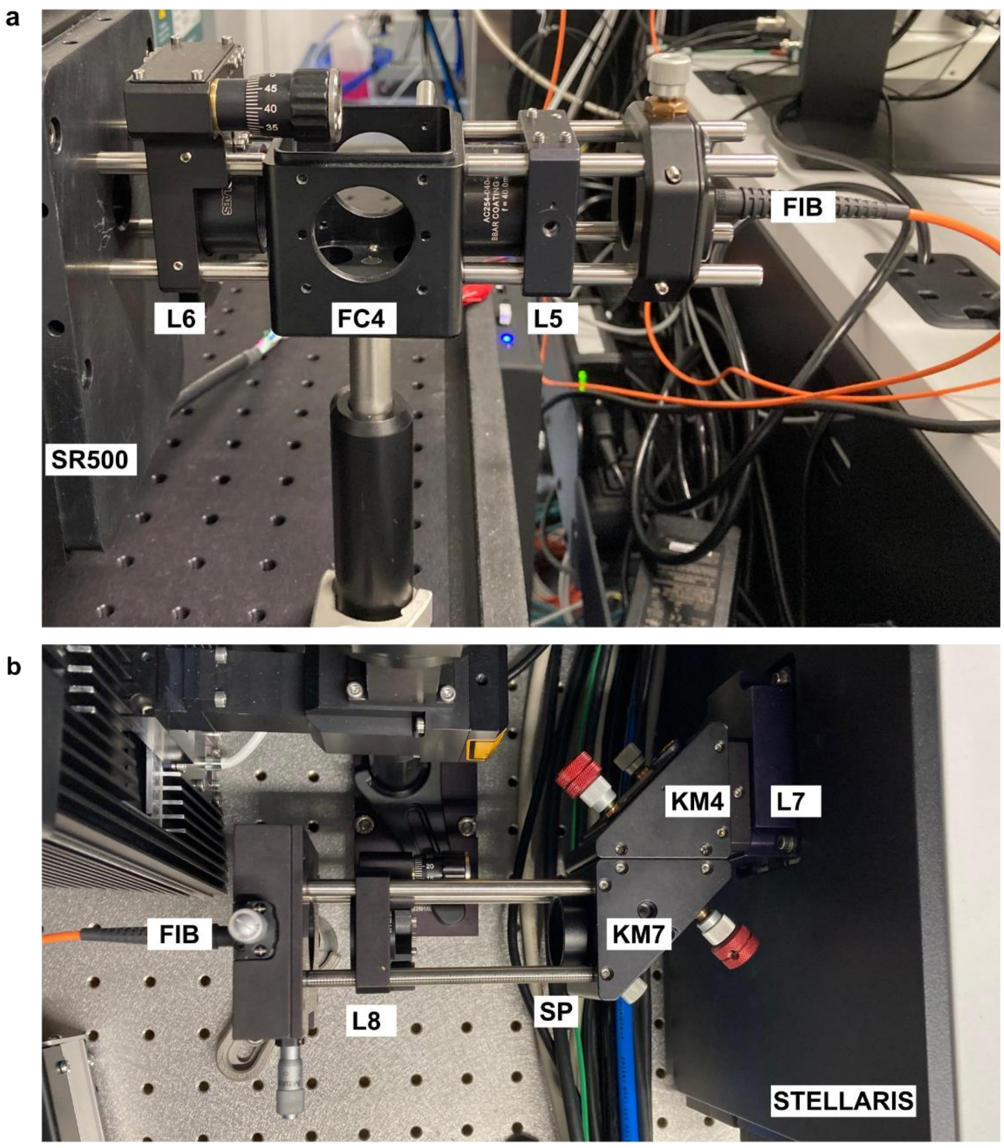


Figure S2 Annotated photographs of the optical fiber mounting. a, Relay lens system for coupling to the spectrometer and **b**, coupling LP emission from the X1-Port on a Leica Stellaris 8 commercial confocal microscope to the optical fiber.

The relay system is connected to the spectrometer with a cage system adapter plate, which ensures that the relay cage system and the spectrometer share the same optical axis. If such an adapter is not available, the relay system should be manually pre-aligned by measuring the correct optical height with a ruler and by careful visual inspection, before attempting the fine alignment procedure. The two cameras (CCD camera (CAM2), and line-scan camera (CAM3)) used for alignment and detection are connected to the two outputs of the spectrometer. They are never used simultaneously, rather the CCD camera is used to pre-align the fiber; therefore, it is also possible to use a spectrometer with a single output port and to dismount the CCD camera following the pre-alignment. We have designed a custom camera flange to mount the line-scan camera to the output port of the spectrometer (technical drawing available as Supplementary File). This flange consists of 3 adjustable parts, allowing movement in x and z as well as rotational alignment.

Next, we discuss the synchronization and control of timing between the different components. Components not requiring precise synchronization are controlled by their respective software, which allows maximum flexibility to exchange them for components from different manufacturers. This includes the excitation laser, the spectrometer, the motorized microscope stage, and the brightfield camera. We have developed a custom solution for the confocal scan head and the line-scan camera, which require communication and synchronization. Users who wish to replicate our confocal scanning routine can download our custom software, which is designed to be used in conjunction with the hardware listed in this protocol. The confocal scan head consists of the 2D galvo mirrors that are connected to a NI-DAQ and the piezoelectric objective scanner along with its controller, and all its components are controlled with LabView. The LabView script utilizes functions taken from an open-source confocal scanning program[1] to raster-scan the confocal excitation across the sample with a user-defined scan field and moves the objective scanner in accordance with the raster scanning for the acquisition of z-stacks. The LabView script further generates a pulse at each confocal voxel to trigger the acquisition of a spectrum on the line-scan camera. A coaxial cable connects to the trigger output port of the NI-DAQ and can be terminated with two crocodile clips that connect to the two 'External Trigger 1' pins (-, pin 7; +, pin 8) of the cable assembly from the frame grabber board. (For a more permanent installation, a custom connector can be assembled.) The line-scan camera and the frame grabber are connected by two camera link cables, and they are controlled with our custom HyperspectralConfocal executable as well as with the GenICam ComCam software (available from Teledyne e2v). HyperspectralConfocal.exe is used for setting acquisition parameters and for acquiring and saving data. It updates its own configuration accordingly and writes the key acquisition parameters and settings to a text file, which is automatically read by the LabView script to match the scan field size, resolution, and pixel dwell time to the user-defined input. The line-scan camera (CAM3) continuously acquires 2D images, from now on referred to as buffers, that contain sequences of spectra with the spectral information encoded along their horizontal dimension and all spectra of the raster scan (one spatial x-y-plane) along the vertical dimension of the buffer. Separate buffers are allocated for data from different z-planes and timesteps to acquire x-y-z-t-λ datasets. The parameter text file is also manually loaded into the 'Hyper Terminal' of the CommCam software to configure CAM3 and the frame grabber correctly. Operating the camera at speeds up to 125 kHz requires a specific combination of communication settings that need to match in both programs, which is best done by passing the parameter file rather than setting all parameters manually in both programs (An exemplary camera configuration file and a parameter text file are available in our hardware repository. HyperspectralConfocal.exe has additional functionality implemented for convenience, including quick-converting hyperspectral confocal images for a quasi-real time image display, acquiring images as 2D vertically integrated spectra that are displayed in the LabView script, and batch-exporting the series of images as separate files along with a logfile storing all metadata. This logfile can later be read into the processing functions to correctly interpret the axes of the multi-dimensional data set and to calibrate the scan field automatically.

Supplementary Table 1: List of key optical components for the homebuilt confocal microscope

Component	Position (Figure 5 and Figure S1)	Part number of mount	Part number of optics
Corner mirror for excitation	FC1	DFM1B	DFM1-M-P01
Beamsplitter for brightfield imaging	FC2	DFM1B, DFM1T3	BSX10R
Dichroic mirror to separate fluorescence	FC3	DFM1B, DFM1T3	DMLP567R
Scan lens	L1	SM2A33	LSM03-VIS
Tube lens, confocal	L2	LCP08-M	TTL200-A
Tube lens, brightfield	L3	CP36-M	AC254-075-A-ML
Fiber coupling lens	L4	SM1Z, SM1A6	AC127-019-A-ML
Lens 1, spectrometer relay	L5	CP36-M	AC254-040-A-ML
Lens 2, spectrometer relay	L6	SM1Z	AC254-050-A-ML
Longpass filter, brightfield	LP1	CP36-M	FELH0550
Longpass filter, confocal	LP2	CP36-M	FELH0550
Optical fiber	FIB	ST1XY-D-M, SM1SMA (confocal side); CXY1A, SM1SMA (spectrometer side)	M96L02 (‘open pinhole’ fiber) or M65L02 (‘closed pinhole’ fiber)

Supplementary Note 2: Implementation with commercial confocal microscopy

Even though some commercial confocal microscopes offer multichannel spectral detection and thus hyperspectral imaging, their spectral resolution is generally not sufficient to analyze the emission of LPs. Here we describe how an existing commercial confocal microscope can be upgraded with our hyperspectral detection system. We first discuss the requirements the commercial confocal microscope has to meet to be suitable for this. Fundamentally, there are three important prerequisites: (1) The confocal microscope needs to be able to excite the LPs with a pulsed laser source, (2) their fluorescence/lasing signal needs to be accessible, and (3) the scan head needs to provide a suitable trigger signal for synchronization with the line-scan camera.

Excitation of LPs typically require short-pulsed lasers with pulse energies in the pico- to nanojoule regime[2,3] under single-photon excitation, which are typically not integrated in commercial confocal microscopes. However, some microscopes allow to couple alternative lasers into the scan head. Another option that is more widely available is two-photon excitation, as semiconductor nanodisk LPs lase at only a fraction of the power available from the tunable ultra-short pulsed infrared lasers used in commercial multi-photon microscopes[4].

The option to outcouple a signal for analysis with custom detectors is also available for many commercial microscopes. For optimum image quality, it is preferable to use the de-scanned signal.

Lastly, the timing and triggering configuration needs to be compatible with the acquisition parameters of the line-scan camera. For the camera configuration used here, the pixel dwell times on the confocal microscope should be > 8 µs to match the acquisition speed of the line-scan camera, which is especially important when triggering individual lines on the line-scan camera with the 'pixel trigger' of the confocal scan head. Many scan heads alternatively provide a 'line trigger' for a line of pixels in the confocal image. Due to the different data format in the hyperspectral dataset, the 'line trigger' confocal signal can be used as a 'frame trigger' in the hyperspectral image context, where the individual pixels need to be gated by the internal clock of the line-scan camera. A consequence of this change in settings is that buffers now contain only information on one line of spatial pixels, where the spatial y-axis is now stored in separate buffers in addition to the z- and t-dimensions.

Supplementary Table 2: List of optical components of the outcoupling module (commercial microscope)

Component	Position (Figure S2)	Part number of mount	Part number of optics
Collimation Lens	L7	SM1M10, Custom adapter	AC254-100-A
Coupling Lens	L8	SM1Z	AC127-030-A-ML
Short-pass filter	SP	SM1L03	FESH0750
Optical Fiber	FIB	ST1XY-S/M, SM1SMA	M96L02 ('open pinhole' fiber) or M65L02 ('closed pinhole' fiber)

To provide a practical example, we discuss our implementation of confocal hyperspectral imaging with a Leica Stellaris 8 confocal microscope. The Stellaris 8 microscope contains the optional DIVE multi-photon modality (including a Spectra Physics Insight X3 Dual femtosecond excitation laser), the 'X1-port' (on the scan-head) for outcoupling of the de-scanned signal, and the Leica trigger unit providing a set of standard 'trigger out' signals. Here, we use the fixed line of the multi-photon laser (1045 nm) to excite semiconductor nanodisk LPs. To collect the emission from these LPs, a small outcoupling module (Figure S2b) was built to collimate the light exiting the X1 port and couple it into an optical fiber. A custom lens adapter (technical drawing available on our data repository) is connected to the output port for collimation, which forms the base for attaching a cage system that is concentric with the optical path. The fiber coupling setup is similar to the one used in the home-built confocal scan head, including use of the same optical fibers that can be readily exchanged using the flexible fiber adapters. The same relay module (Figure S2a) can therefore continue to be used on the spectrometer side. To ensure correctly timed acquisition, the 'line trigger' signal from the confocal scan head is connected as 'frame trigger' to the frame grabber, such that individual line-scan camera buffers are triggered to contain a line of spatial pixels as described above. The line-scan camera triggers individual spectra ('lines') internally, therefore the line integration time is manually set to match the pixel dwell time of the confocal scan. The dimensions of the scan field are also manually matched such that the height of an individual line-scan camera buffer is equivalent to the number of pixels in x in the image obtained with the Leica software, and the number of buffers accounts for the remaining dimensions (y*z*t). The resulting organization of buffers is therefore different compared to the pixel trigger configuration described in the main part of this protocol and requires an adaptation of our processing functions (Supplementary Code 4) to visualize the resulting images (Figure S3).

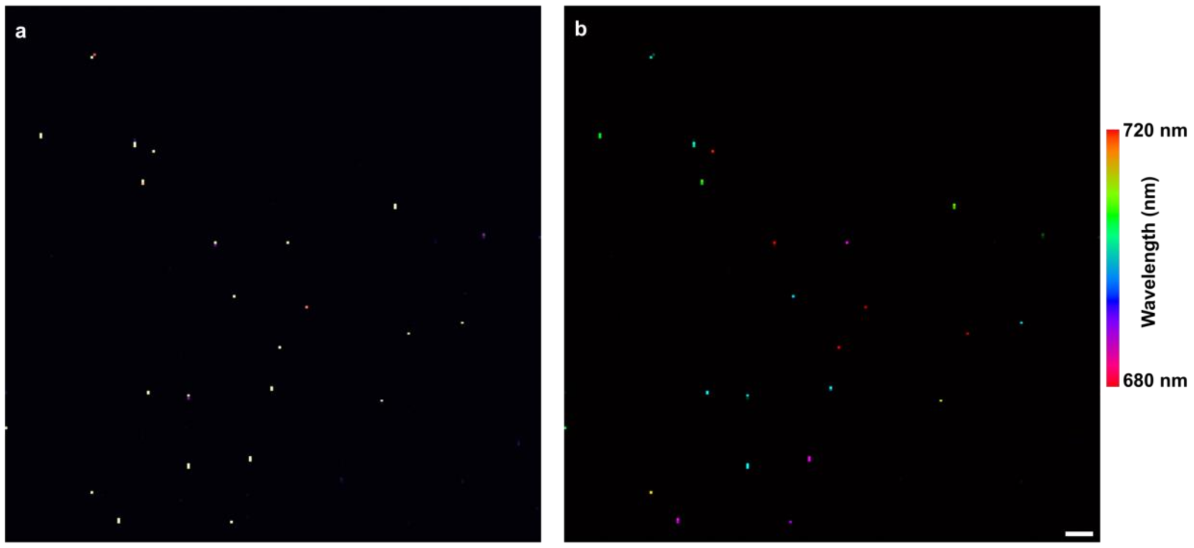
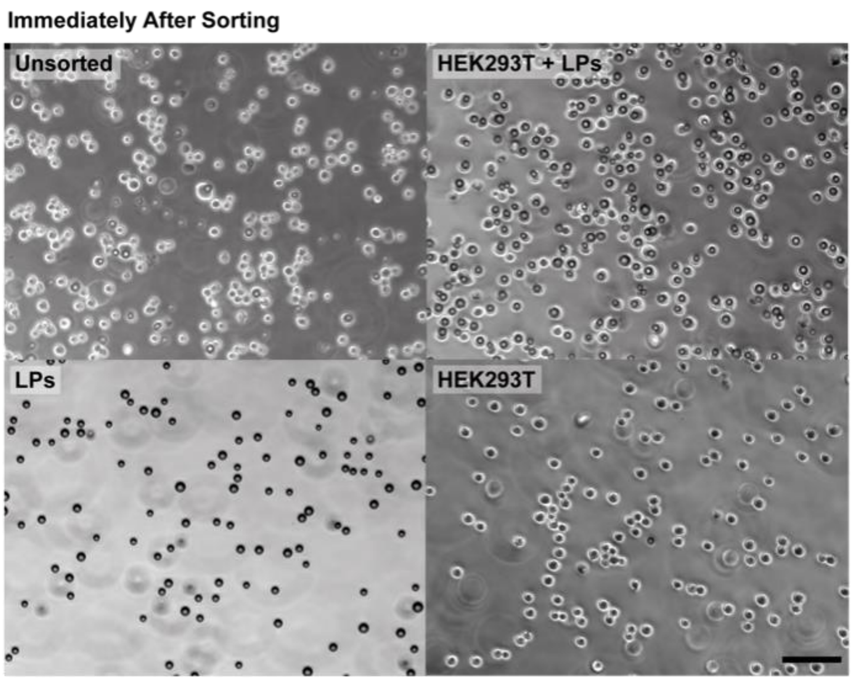


Figure S3 Hyperspectral confocal image of semiconductor nanodisk LPs obtained on a modified Leica Stellaris commercial microscope. **a**, intensity color coding, and **b**, custom wavelength LUT. Scale bar, 25 µm.



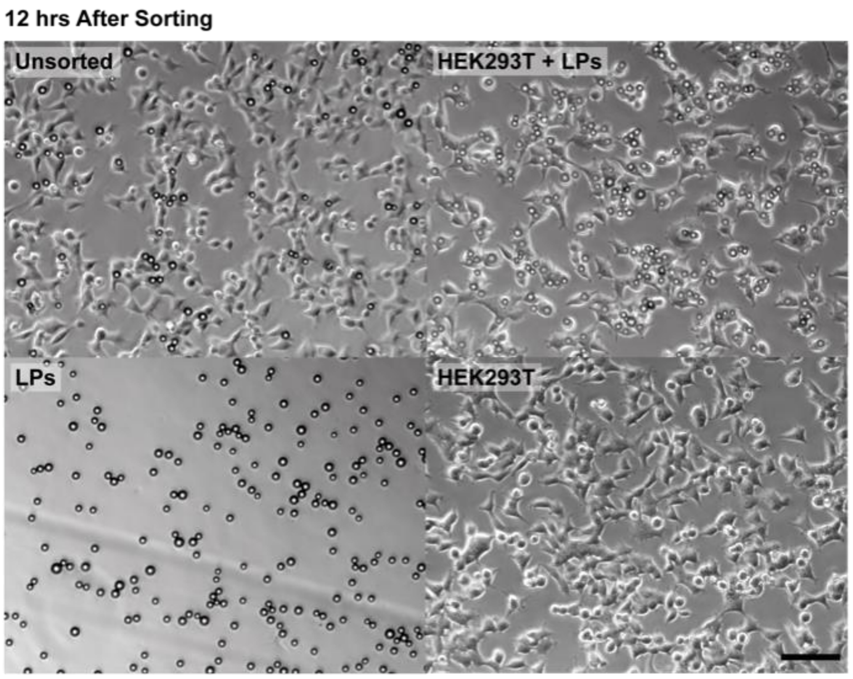


Figure S4 DIC microscopy images showing cells prior to FACS sorting (unsorted) and the three sorted populations (LPs, HEK293T+LPs, HEK293T), immediately after FACS (top) and after reattachment (bottom). Scale bars, 100 µm.

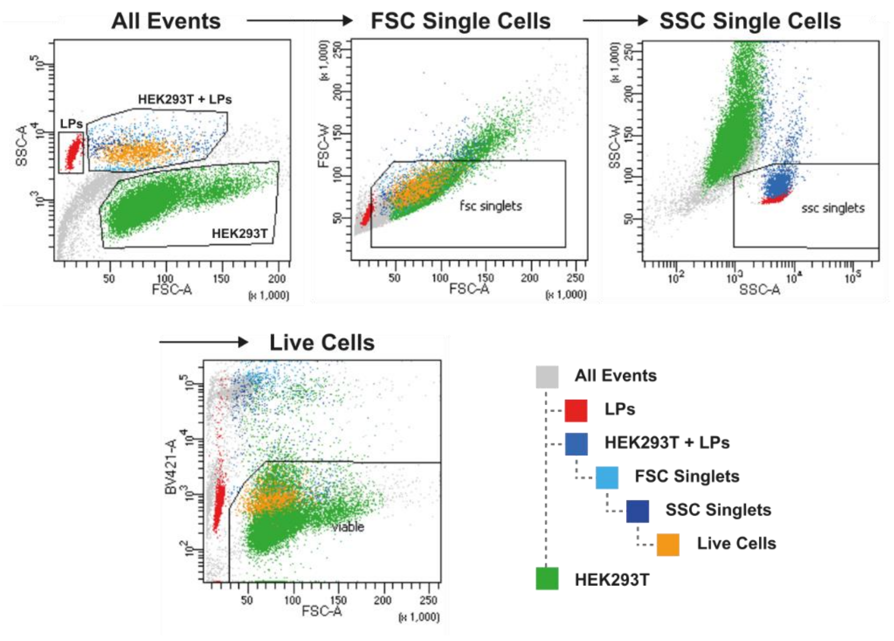
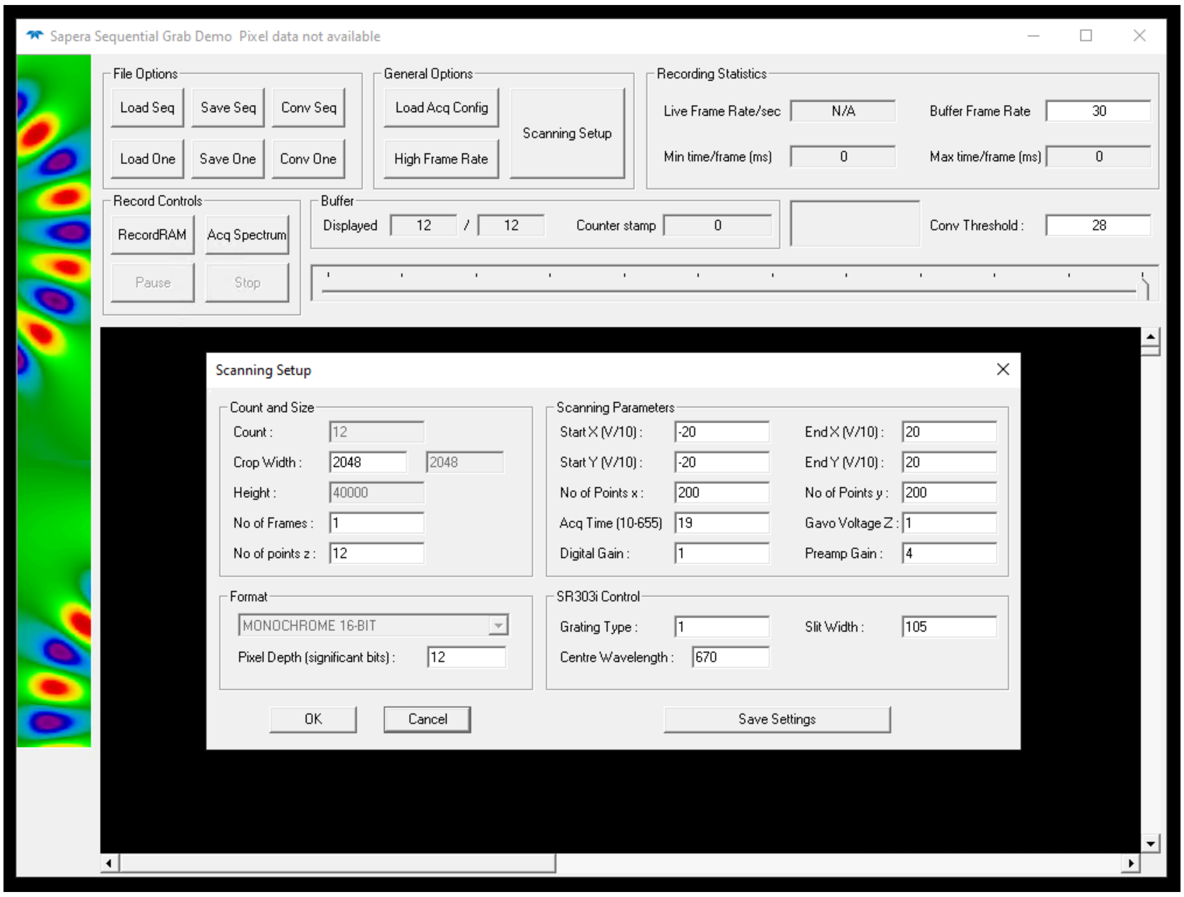


Figure S5 FACS gating strategy. Cell sorting was performed using a BD FACSFusion with FACSDiva 8.0.1 software. Cells were sorted at RT using a 130 µm nozzle and sheath pressure was set at 14 psi. 0.9 % NaCl was used as sheath fluid. Gating was based on forward (FSC-A) and side scatter (SSC-A) distribution, identifying three populations (LPs, HEK293T, HEK293T+LP). Agglomerates were identified from the FSC-W and SSC-W signals. DAPI (BV421-A) was used for dead cell exclusion, which was excited by a 405 nm laser and detected using a 450/50 nm bandpass filter. PE-A (LP emission) and the ImageStream (imaging cytometer) can further be used for validation.


Sapera Sequential Grab Demo Pixel data not available

File Options: Load Seq | Save Seq | Conv Seq | Load One | Save One | Conv One

General Options: Load Acq Config | High Frame Rate | Scanning Setup

Recording Statistics

Live Frame Rate/sec	N/A	Buffer Frame Rate	30
Min time/frame (ms)	0	Max time/frame (ms)	0

Record Controls: RecordRAM | Acq Spectrum | Pause | Stop

Buffer: Displayed 12 / 12 Counter stamp 0

Conv Threshold : 28

Scanning Setup

Count and Size

Count :	12	
Crop Width :	2048	2048
Height :	40000	
No of Frames :	1	
No of points z :	12	

Scanning Parameters

Start X (V/10) :	-20	End X (V/10) :	20
Start Y (V/10) :	-20	End Y (V/10) :	20
No of Points x :	200	No of Points y :	200
Acq Time (10-655)	19	Gavo Voltage Z :	1
Digital Gain :	1	Preamp Gain :	4

Format: MONOCHROME 16-BIT
Pixel Depth (significant bits) : 12

SR303i Control

Grating Type :	1	Slit Width :	105
Centre Wavelength :	670		

OK | Cancel | Save Settings

Figure S6 Screenshot of the HyperspectralConfocal GUI. The 'Scanning Setup' menu is opened where all parameters for the acquisition can be entered. Once confirmed with 'ok', they are communicated to LabView and written to a log file. The software will also automatically update its own configuration file, which is confirmed by clicking 'ok' again. The buttons under 'Record Controls' are used for image and spectral acquisition, and the 'File Options' menu allows (batch)-conversion and exporting of image files. For the conversion functionality, the color scaling and thresholding of the converted images can be adjusted by setting the 'Conv Threshold' to an even number, representing the minimum counts per pixel, above which the corresponding pixel is colored-in in the hyperspectral images.

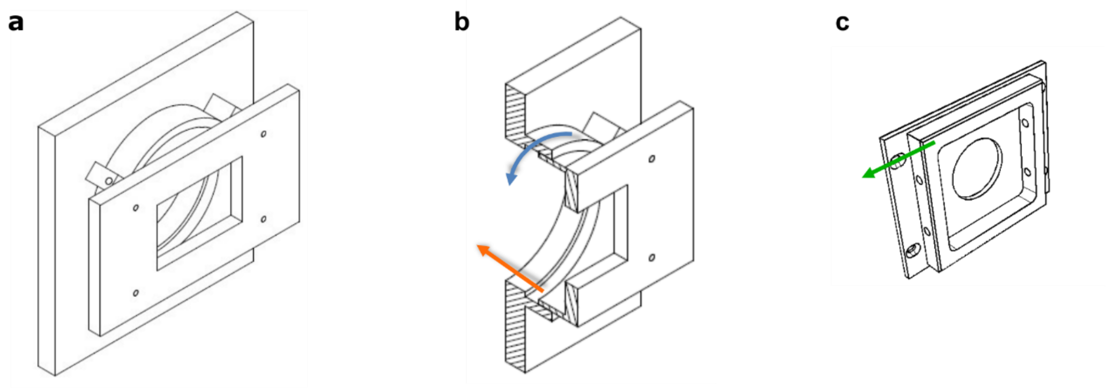


Figure S7 Line-scan camera adapter, technical drawing of **a**, the assembled part and **b**, a cross-section, indicating alignment directions allowed by physical movement of the detector flange for adjustment in z (orange arrow) and rotational alignment (blue arrow). **c**, The x-direction (green arrow) is aligned using elongated screw taps on the line-scan camera holder plate, which is attached to the adapter shown in a, b.

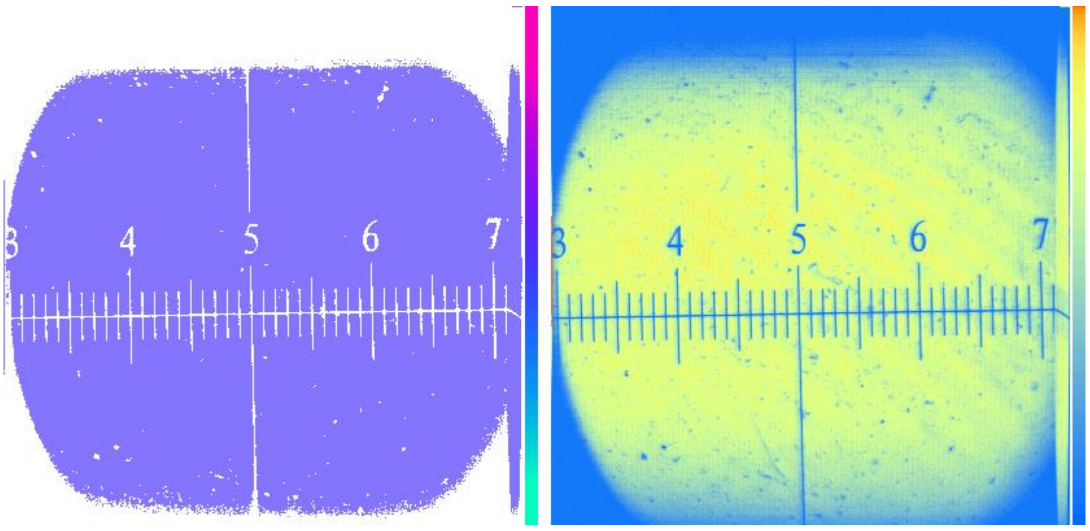


Figure S8 Typical hyperspectral confocal image converted by the hyperspectral confocal GUI, here showing the calibration slide acquired with a 10× objective and a +- 50 V scan field size. The images are color-coded by wavelength (left) and intensity (right) with the intensity scaling determined by the 'threshold' value in the HyperspectralConfocal software. Minor clipping and shadowing artefacts are observed on the edge of this image.

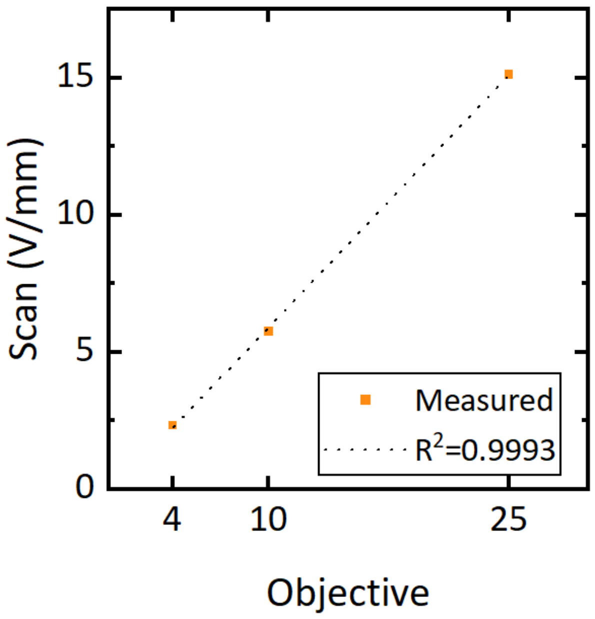


Figure S9 Calibration of the scan field for different objectives, with the measured scaling factors of the galvo voltage applied per mm size of the field-of-view (orange). A linear fit recovers the expected relationship between objective- and scan-field magnification (black dashed line).

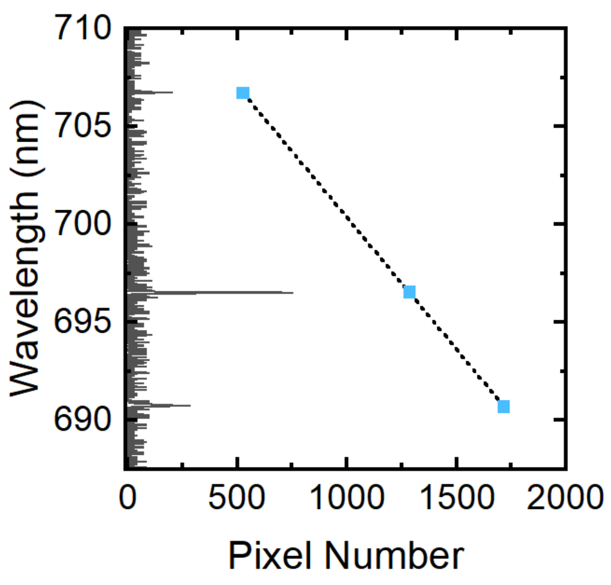


Figure S10 Spectral calibration of the line-scan camera, using three well defined peaks in the measured spectrum (grey) to calibrate pixel numbers to the known wavelengths of the calibration lamp peaks (light blue). A linear fit (dashed line) gives the calibration function.

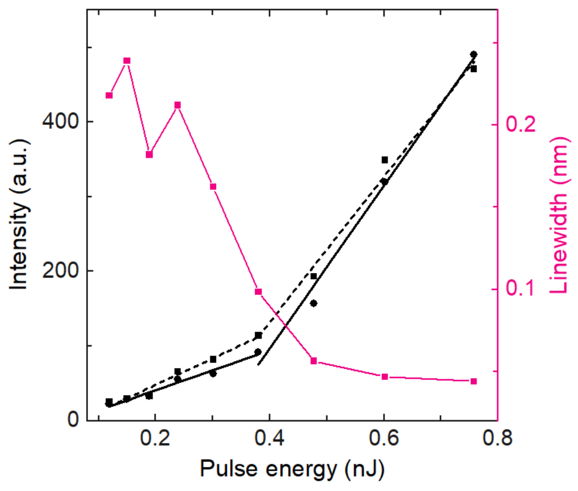


Figure S11 Lasing threshold of a FluoRed bead (black curve), showing a threshold of approximately 0.4 nJ, and linewidth continuously narrowing until a resolution-limited lasing peak is observed above threshold (magenta curve). Two threshold curves were recorded while going from low to high pulse energies (squares, dashed line) or high to low pulse energies (circles, solid line).

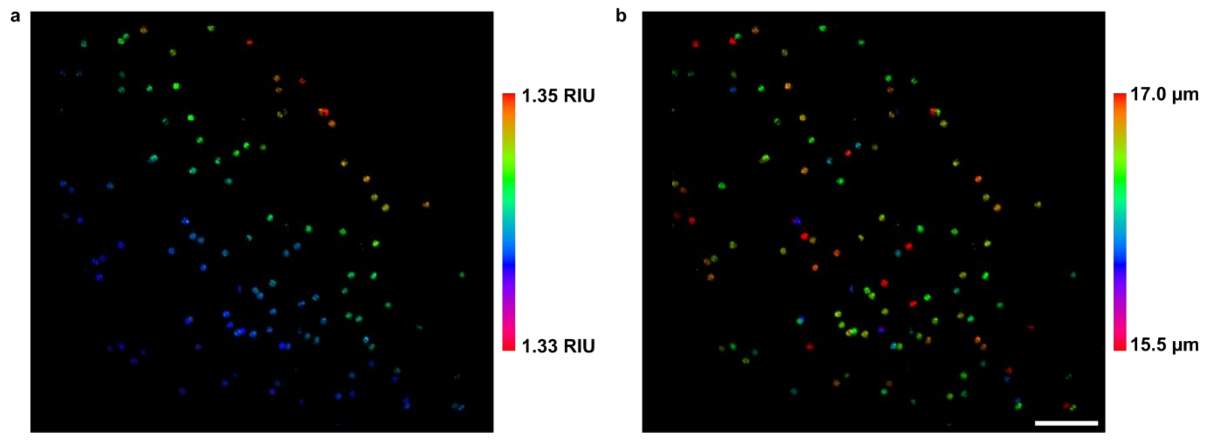


Figure S12 Hyperspectral confocal image of FluoRed microbead LPs dispersed in agarose, maximum intensity projection (MIP). Color-coded according to the results from the asymptoticExpansion Matlab script for **a**, external refractive index and **b**, bead diameter. Scale bar, 200 µm.

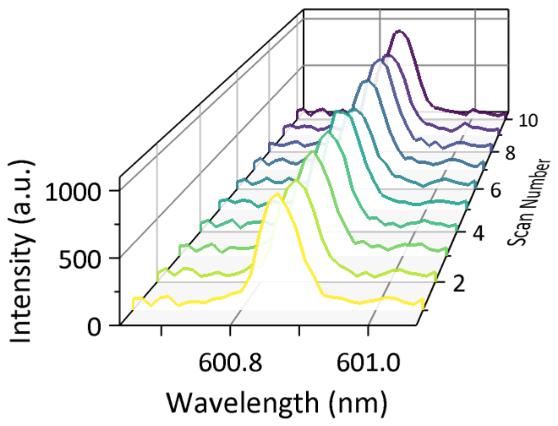


Figure S13 Stability of FluoRed microbead LPs during consecutive acquisition of hyperspectral images, recorded with a 10× objective (NA=0.45).

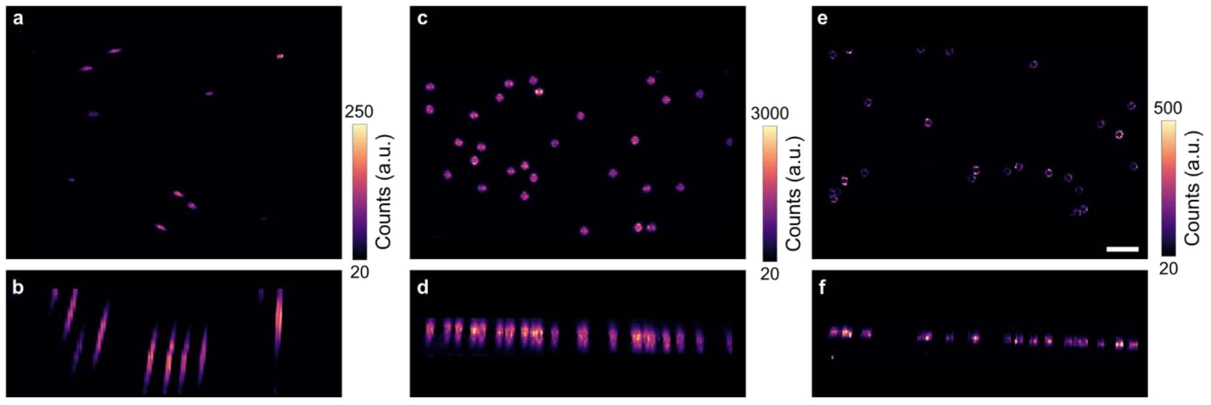


Figure S14 Axial resolution and quality of hyperspectral confocal images of FluoRed microbead LPs. **a, b**, Images with the back aperture of the objective underfilled and an the “open pinhole” fiber, **c, d**, the back aperture overfilled and the “open pinhole” fiber, and **e, f**, the back aperture overfilled and the “closed pinhole” fiber. Images are displayed as x-y MIPs (top row) and x-z MIPs (bottom row). Scale bar, 100 µm.

Supplementary Note 3: Software Architecture

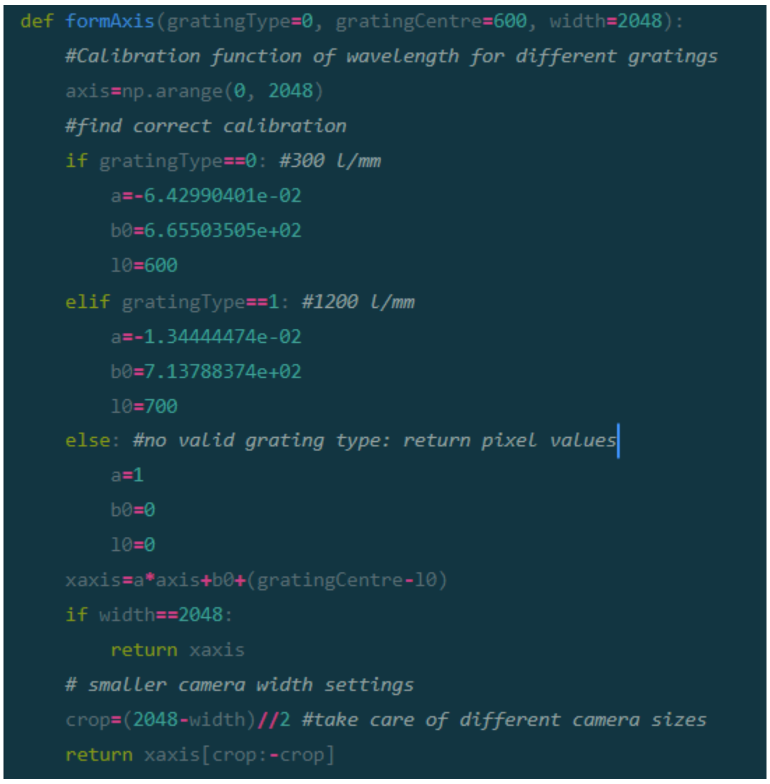
Our software processing workflows are predominantly written in Python. Here, we provide four Supplementary Code Files (as jupyter notebooks) and three python helper files that contain all processing functions and are imported at the beginning of each Supplementary Code File. The four Supplementary Code Files serve as application examples to demonstrate processing functions and algorithms for a range of different applications. Supplementary Code 1 explains fundamental operations, including organizing the multi-dimensional datasets into correctly formatted images, 3D-stacks, and sequences of images or stacks. It also explains how to extract and store individual spectra from specified positions in the image. In Supplementary Code 2, we discuss our treatment of multi-mode spectra, with an emphasis on refractive index sensing. This workflow includes high-throughput peak fitting to extract the external refractive index around each LP and its size from individual spectra. The results can be displayed as simple 2D traces, or as images with custom color-coding in the open-source interactive viewer napari[5]. Supplementary Code 3 pre-processes hyperspectral images of single-mode LPs to a data format compatible with the open-source ND particle tracking algorithm trackpy[6]. Information of the automatically tracked LPs is then organized into a custom data structure that allows displaying trajectories of LPs, the evolution of their mode positions, and other quantities of interest. We also include Supplementary Code 4, where our workflows were adapted to data acquired in a different format (due to a different trigger configuration), which applies to the adaptation of our protocol for integration with commercial confocal microscopes. Each of the files provided in our repository is explained in more detail below and can be found at https://github.com/GatherLab/sphyncs[7].

"utils.py"

The python helper file "utils.py" first imports the main scientific computing libraries used in our processing toolbox (numpy, matplotlib, scipy, imageio, napari, math). "utils.py" contains a few useful functions including mathematical expressions, spectral analysis, and elementary image data handling. These functions are automatically called as part of larger algorithms without needing further optimization or adaptation. An exemption is the spectral calibration function 'formAxis', which must be updated after acquiring the calibration data during Procedure II of the protocol. 'formAxis' transforms pixel number p on the line-scan camera into wavelength λ, assuming a linear mapping:

$$\lambda(p, \lambda_c) = a \times p + b_0 + (\lambda_c - \lambda_{c0}),$$

where a and b_0 are determined from fitting a linear function to the measured spectral calibration data, and λ_{c0} and λ_c are the center wavelength of the spectrometer grating during the calibration measurement and during subsequent measurements, respectively. To find a and b_0, the ground truth peak positions (λ) should be first estimated from the reference spectrum measured on a calibrated CCD camera, and then refined by comparing them to the closest matching atomic transition line of the calibration lamp. The corresponding pixel values p should be determined using a gaussian fit for each peak. At least three well-defined peaks should be used to fit a linear function (Figure S10), from which the slope (a) and intercept (b_0) are stored and updated in the calibration function. λ_{c0} also needs to be included in the calibration function, such that the spectral calibration is valid for any center wavelengths λ_c (specified as input argument 'gratingCentre' when using 'formAxis'). Note that the accuracy of this linear approximation might be limited for very extended wavelength ranges, therefore the calibration measurement should include peaks in the same range of the LP emission, or as close as possible. Lastly, we have encoded a case structure in our calibration function, allowing to store multiple calibrations for different gratings (Figure S15).



```python
def formAxis(gratingType=0, gratingCentre=600, width=2048):
    #Calibration function of wavelength for different gratings
    axis=np.arange(0, 2048)
    #find correct calibration
    if gratingType==0: #300 l/mm
        a=-6.42990401e-02
        b0=6.65503505e+02
        l0=600
    elif gratingType==1: #1200 l/mm
        a=-1.34444474e-02
        b0=7.13788374e+02
        l0=700
    else: #no valid grating type: return pixel values
        a=1
        b0=0
        l0=0
    xaxis=a*axis+b0+(gratingCentre-l0)
    if width==2048:
        return xaxis
    # smaller camera width settings
    crop=(2048-width)//2 #take care of different camera sizes
    return xaxis[crop:-crop]
```

Figure S15 Screenshot of the spectral calibration function. The parameters a and b0 are updated with slope and intercept of the linear fit of the calibration measurement, and the center wavelength used during the reference measurement is stored as l0. The calibration function can handle multiple gratings, and cases where only part of the camera was read out.

“acquisitionExtended.py”

This file is the acquisition class definition, i.e., a custom Python object specifically designed to represent hyperspectral confocal measurements. This data structure can store all relevant measurement settings as attributes (Supplementary Table 3), which can be automatically set by loading the measurement log file or be user-defined. The acquisition class contains all data processing methods (i.e., functions that are specific to the data class). The main advantage of this approach is that operations that require knowledge of important acquisition attributes, which includes almost all processing functions, can access them directly from the acquisition object, rather than requiring user input. These processing methods are used in the Supplementary Code Files and will be explained in detail later.

Supplementary Table 3: Attributes of the Acquisition Data Class

Attribute	Explanation
name	The name of the stored .tif files of the measurement, without the ‘framexxxxx’ extension.
directory	Path of the directory containing the raw data. This can be a folder on external drives, too, which can be preferable due to the large file sizes.
xDim, xDim	Number of spatial pixels in x, y. The height of an individual camera buffer will equal xDim × yDim.
zDim	Number of z-slices in the stack
tDim	Number of frames in a x-y-(z-)t sequence. The total number of buffers acquired during the measurement will equal zDim × tDim.
xStep, yStep, zStep	The size of individual voxels (x, y, z), in µm.
tStep	The duration between each timestep.

xaxis	Array containing the calibrated wavelength-values corresponding to each camera pixel along its spectral axis.
wLDim	The width of individual camera buffers, equivalent to the number of data points on the spectral axis. This will be equal to the 2048 pixels of the line-scan camera, unless cropped to a smaller number.

Supplementary Code 1

Here, we first import “acquisitionExtended.py” along with its dependencies, including “utils.py” and the scientific computing libraries used (see jupyter notebook, Cell 1, line 1). We next define the directory and name of the data set and the corresponding log file (Cell 2, lines 1-3), before calling the constructor of the acquisition class (Cell 2, line 4). With the log file name as one argument and the size of the scan field at ± 50 V as the second argument, the ‘autoCalibrate’ method can be used (Cell 3, line 2). It will retrieve all relevant settings from the log file, store them as attributes in the acquisition object, and use them to calculate the correct voxel size and spectral calibration. Where no log file is present, the acquisition attributes can be specified manually when calling the class constructor (Cell 4, line 3), which requires subsequent manual calculation of the voxel size (Cell 4, lines 5-6). To process an individual image, the ‘construct2D’ method is used (Cell 5, line 7). The timestep and z-position of the 2D image need to be specified, and there are a few additional optional arguments. With ‘threshold’, only pixels above a given threshold are included in the image formation; enabling ‘save’ automatically saves the processed images; s1 and s2 allow limiting the bandwidth of the color-mapping in spectral images to a user-defined range (useful for avoiding mode jumps in color-coding of multi-mode spectra); with show=’True’, the processed images will automatically be displayed; ‘specImg=True’ enabled spectral processing in addition to intensity-based images; and lastly, a custom scale value ‘scaleVal’ can be entered to limit the color map of the intensity image to a certain range. The function returns an intensity-based image ‘imap’, and the spectral image ‘maxmap’ (if spectral processing was enabled). Individual spectra can be extracted with the ‘getSingleSpectrum’ or ‘getSingleSpectrumIntegrated’ method (Cell 6). A position (x, y, z, t) needs to be specified, and, if the resulting spectra should be exported as .txt files (save=’True’), a file path for the saved spectra also needs to be passed to the function. ‘getSingleSpectrum’ retrieves the spectrum exclusively from the designated position, whereas ‘getSingleSpectrumIntegrated’ returns an average of all spectra within a certain tolerance (keyword argument ‘tol’, given as numbers of pixels) that are above a certain threshold (keyword argument ‘thresh’).

Stacks are processed with the ‘construct3D’ (intensity only) and ‘constructSpec3D’ (intensity and spectral information) methods (Cell 7, line 1). The keyword arguments of this method are analogous to its 2D equivalent. Before loading the images in napari (Cell 8, line 6), we typically define a viewing angle of the stack (Cell 8, line 2), the correct spatial scaling (Cell 8, line 3; directly taken from the acquisition object), and an appropriate intensity scaling. Stacks can also be color-coded according to their spectral information, which requires creating separate RGBA color channels from the image data (Cell 9, line 2) with the ‘VAtoRGBA’ function (inputs: the spectral image, the intensity image, and the lower and upper limit of the color map). The resulting color-arrays can then be loaded into napari as separate channels (Cell 9, lines 6-8). The 3D-rendered images can be exported as screenshots from napari, where a simple loop over a range of angles can be used to create a flyover video (Cell 10).

Supplementary Code 2

To begin, we set up the acquisition object (Cell 1, lines 1-16; as explained for Supplementary Code 1). Next, the ‘fitSpectraToFiles’ method is called for detecting all peaks in the data set and storing them in a data format suitable for applying the mathematical model. ‘fitSpectratoFiles’ will analyse all pixels above a certain intensity threshold (keyword argument ‘thresh’) and detect all peaks above a certain height (keyword argument ‘height’) in each spectrum, assuming a given spacing between adjacent peaks (keyword argument ‘tol’, in pixel number), using an inbuilt scipy function. All detected peaks are then fitted with a gaussian function, where the optimization algorithm looks at a subset of the whole spectrum, which is a window of size ‘tol2’ (in pixel numbers) around the peak. A minimum number of peaks (‘noPeaks’) needs to be found such that spectra are included in the result files. The result files contain separate columns for each spectrum, where the position in the image is stored, as well as the intensity of the spectrum and the fitted peak positions. Such data files can be analysed using the ‘AsymptoticExpansion’ MATLAB script, which will be explained in more detail later. The script appends the fitted external refractive index (in RIU), LP size (in µm), and residual error of the fit (in m) to the end of each result file. These values can again be read into python for further processing and displaying with the ‘ResultsFromFiles’ method. Either the refractive index (param=’n’; Cell 3, line 2), the LP size (param=’d’; Cell 3, line 3), or two consecutive lasing peaks (param=’peak’; Cell 3, line 4) can be read from the files, and for either option, a critical residual error (‘maxRes’) needs to be specified to exclude poor fits (typically, we use a number similar to the spectral resolution of the measurement, i.e. 4×10^{-11} m). If LPs remain stationary during the measurement, their positions can be used to extract traces of the quantities of interest and plot them (Cell 4). The results can also be displayed as 3D-rendered projections in napari, using the ‘cmapResultsFromFiles’ method. In addition to the parameter of interest, and the critical residual error, upper and lower bounds of the color mapping need to be included. The refractive index calibration data can be processed following the same routine (Cell 7), to determine the size distribution of the LPs and their internal refractive index (Cell 8).

<u>**Asymptotic Expansion (MATLAB)**</u>

This optimisation workflow compares the measured peaks to a theoretical model for the resonance positions of whispering gallery modes based on the asymptotic expansion of resonance frequencies in Mie scattering[8]. Our optimisation assumes that we only observe the fundamental radial mode number *l*, therefore the measured mode position λ depends on LP size *d*, refractive indices of the LP (n_{int}) and of its environment (n_{ext}), the angular mode number *m*, and the polarization mode number *p*. For each spectrum containing a pre-defined minimum number of lasing peaks (‘noPeaks’ in Supplementary Code 2), the optimization seeks to find a solution for the LP size, the external refractive index, and the angular mode numbers of consecutive modes. A range of each of these three parameters therefore needs to be defined prior to using the optimisation function (‘fit_peaks.m’). The internal refractive index is assumed to be constant, where we either use the literature value, or find a more accurate number as part of our refractive index calibration workflow (see below). Using ‘te_or_tm.m’, the algorithm first automatically finds the polarization mode numbers. Next, the number of initial guesses is specified (we typically use 100) to begin the optimization. The ‘spectral_peaks.m’ function is called to calculate a set of mode positions based on the asymptotic expansion model (‘wgm_schiller.m’), and the objective of the optimisation is to minimise the least square error between the calculated and experimental mode positions. Finally, the optimized values for LP size, the external refractive index, and the angular mode numbers are found, and a residual error for each fit is determined. These values are stored in the result files as described above.

To improve the accuracy of this approach, we typically run a calibration measurement that optimizes the internal refractive index for a known external refractive index, using ‘Cali_internalref.m’ and ’fit_peaks_internalref.m’. Here, we use the calibration data set of LPs in a medium of homogenous and well-known refractive index (like deionised water). The optimisation works analogous like the one described for finding the external refractive index, using the same functions (‘wgm_schiller.m’, ‘te_or_tm.m’, and ‘spectral_peaks.m’). After the optimisation, users can filter the results for a critical residual error. From these, the script calculates the average internal refractive index, which can be used as an input parameter for the internal refractive index when using ‘fit_peaks.m’ and returns a .txt summary file with the filtered results.

Supplementary Code 3

Here, we first create and calibrate the acquisition object, allowing batch-processing the image files into a data format compatible with the ND-tracking algorithm trackpy (Cell 1). The ‘batchProcessND’ method is called after deciding whether binning should be applied to reduce the file size (‘nBin’ keyword argument) and defining a range of intensity values to use during the tracking (‘upLim’ and ‘lowLim’; Cell 1, line 14). In this step, the wavelength-axis is removed from the individual buffers and treated analogous to the z-axis (i.e., encoded in separate images). All images are stored in a .zip file (Cell 2) that can be read into the ‘trackpy’ algorithm using the ‘pims.ImageSequenceND’ function (Cell 3, line 1). Next, ‘tp.batch’ detects features in the whole data set, expecting features to have a size (in pixels) equal to the input argument ‘diameter’ (Cell 3, line 6). After calibrating the sequence, using the step sizes from the Scan object (Cell 4, lines 1-6), detected objects can be linked with the ‘tp.link_df’ function, and the results are stored as a .csv file. To organize the tracking results into a library of LPs, we make use of two additional data structures: the ‘disk’ class for individual LPs, and the ‘trackDataND’ class for data from the entire measurement. The results from the tracking algorithm are loaded into the database by calling the class constructor, passing the acquisition class an input argument for correct calibration of all dimensions (Cell 5, line 2). The data base contains all tracked LPs in a library of ‘disk’ objects, and it can display trajectories of all LPs in a common image, either color-coded by wavelength (Cell 6, line 2) or by time (Cell 6, line 3), after defining a minimum length to exclude very short trajectories. One can also investigate individual LPs (Cell 7) by first extracting them from the database, and then using their inbuilt methods ‘plotTraj’ to plot the evolution of any quantity of interest (to be specified as keyword argument ‘ax0’) over the duration of the measurement, or to plot individual x-y-migration paths with the ‘plotXY’ method (analogous to that of the whole database, but for one LP only).

Supplementary Code 4

This code file applies to data acquired with the ‘Frame Trigger’ configuration, which we recommend for integrating hyperspectral measurements with commercial confocal microscopes. The different triggering configuration leads to a different data format, which requires adaptations when processing the data. Specifically, individual buffers contain only one spatial dimension (x) here, in addition to the spectral information, and the additional axes (y-z/t) are all encoded as separate buffers. We have implemented additional ‘CCF’ (commercial confocal microscope) versions of some basic processing methods, which account for this difference, but take the same input and return the same results as the ‘regular’ processing methods. The data sets acquired with a commercial microscope can be processed with the same acquisition data class, but the ‘autoCalibrateCCF’ method is used for calibrating the scan from the log file (Cell 2), and the ‘constructSpec2DCCF’ method can construct intensity- and spectral images (Cell 3), which can be color-coded and displayed in napari (Cell 4). Individual spectra from the data set can also be extracted from defined positions using the ‘getSingleSpectrumCCF’ method (Cell 5). Lastly, batch peak fitting was also implemented for the commercial confocal microscope case with ‘fitSpectraToFilesCCF’ (Cell 6). The resulting data format from this fit is identical to the version for the home-built microscope, ensuring compatibility with further data processing workflows.

References

1. Ferrand, P. GPScan.VI: A general-purpose LabVIEW program for scanning imaging or any application requiring synchronous analog voltage generation and data acquisition. *Comput Phys Commun* **192**, 342–347 (2015).
2. Fikouras, A. H. *et al.* Non-obstructive intracellular nanolasers. *Nat Commun* **9**, 4817 (2018).
3. Dannenberg, P. H. *et al.* Multilayer Fabrication of a Rainbow of Microdisk Laser Particles across a 500 nm Bandwidth. *ACS Photonics* **8**, 1301–1306 (2021).
4. Titze, V. M., Caixeiro, S., Di Falco, A., Schubert, M. & Gather, M. C. Red-Shifted Excitation and Two-Photon Pumping of Biointegrated GaInP/AlGaInP Quantum Well Microlasers. *ACS Photonics* **9**, 952–960 (2022).
5. Sofroniew, N. *et al.* napari: a multi-dimensional image viewer for python. (2022) doi:10.5281/zenodo.3555620.
6. Daniel B Allan, Thomas Caswell, Nathan C Keim, Casper M van der Wel & Ruben W Verweij. soft-matter/trackpy: Trackpy v0.5.0. (2021) doi:10.5281/zenodo.3492186.
7. Vera M Titze, Soraya Caixeiro, Marcel Schubert & Malte C Gather. GatherLab/sphyncs. (2023) doi:10.5281/zenodo.8121099.
8. Schiller, S. Asymptotic expansion of morphological resonance frequencies in Mie scattering. *Appl Opt* **32**, 2181 (1993).