Supplementary Information (SI) for Lab on a Chip.

Supporting Information

MagSity Platform: A Hybrid Magnetic Levitation-Based Lensless Holographic Microscope Platform for Liquid Density and Viscosity Measurements

Oyku Doyran Ince[1], *H. Cumhur Tekin*[1,2*]

[1]Department of Bioengineering, Izmir Institute of Technology, Izmir 35430, Türkiye

[2]METU MEMS Center, Ankara 06520, Türkiye

*Corresponding Author E-mail: cumhurtekin@iyte.edu.tr

MagSity Platform

The MagSity platform (**Figure 1A**) is composed of two N52-grade Neodymium (NdFeB) magnets with dimensions of 2 (w) × 5 (h) × 50 (l) mm^3 and magnetic polarization through its height (Shenzhen Aim Magnet Co. Ltd., Guangdong, China), a microcapillary channel (1×1×50 mm^3) (Vitrocom, U.S.A.) and lensless holographic microscopy elements including (i) a light source (3 mm White LED, 12383, Robotistan, Turkiye), (ii) a pinhole with a diameter of 150 µm (Edmund Optics, USA), and a CMOS image sensor (Sony IMX219, Raspberry Pi camera v2, Robotistan, Turkiye). The CMOS-to-microcapillary channel distance is 1 mm, whereas the microcapillary channel-to-pinhole distance is 50 mm and the LED-to-pinhole distance is 20 mm. The magnets are positioned with a fixed distance of 1.7 mm between them and their same poles face each other. The platform was 3D printed via Ultimaker 2+ Connect using 0.4 mm Generic Polylactic acid (Generic PLA). All components were subsequently assembled within the platform having a size of 43 (w) × 120 (h) × 46 (l) mm^3 (**Figure S1**). A mini-computer, Raspberry Pi Model B+ (Int-el International GmbH, Turkey) was used to capture hologram images through CMOS and supply power to the LED driver circuit.

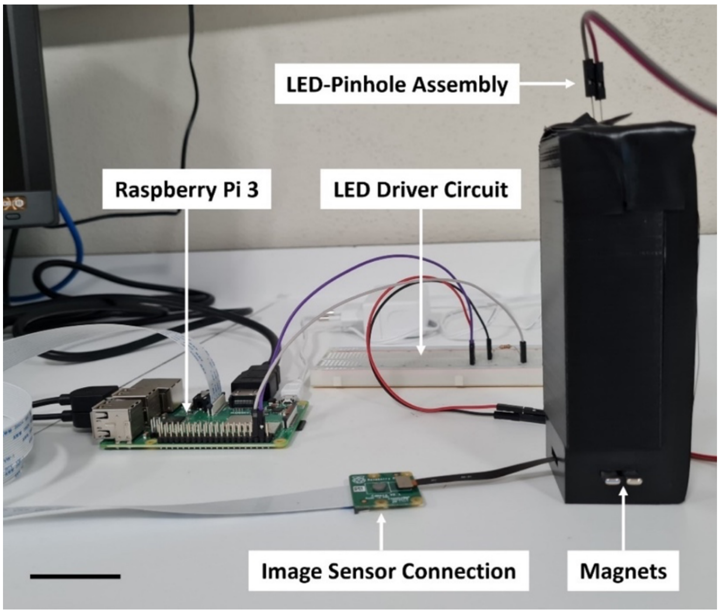

Figure S1. Photograph of MagSity platform. The LED driver circuit consists of a simple 220 Ω current-limiting resistor, which controls the current flow through the LED connected in series. This circuitry is powered with 5V supplied through Raspberry Pi. Scale bar, 3 cm.

Modeling of MagSity Platform

In the platform, particles spiked in the paramagnetic solution tend to move toward the midpoint between the two magnets, where the magnetic induction is at its minimum (**Figure S2**). During this movement, magnetic ($\boldsymbol{F_m}$), drag ($\boldsymbol{F_d}$), buoyant ($\boldsymbol{F_b}$) and inertia ($\boldsymbol{F_i}$) forces act on the microparticle:

$$\boldsymbol{F_m} + \boldsymbol{F_d} + \boldsymbol{F_b} = \boldsymbol{F_i} \tag{S1}$$

In the viscosity measurement configuration (**Figure 1B**), the particles move inside the solution until the acting $\boldsymbol{F_m}$ becomes zero. During this translocation, $\boldsymbol{F_d}$ acts in the opposite direction of the movement. Since $\boldsymbol{F_b}$ is vertical to these forces, it does not affect the particle's movement in this configuration. Due to negligible significance, the inertia forces on microparticles are neglected.[1] Hence, the force balance equation of the particles is expressed as follows,

$$\boldsymbol{F_m} + \boldsymbol{F_d} = 0 \tag{S2}$$

The magnetic force is calculated as,

$$\boldsymbol{F_m} = \left(\frac{V\Delta\chi}{\mu_0}\boldsymbol{B}.\nabla\right)\boldsymbol{B} \tag{S3}$$

, where V is the microparticle volume, $\Delta\chi$ is the magnetic susceptibility difference between the microparticle and the medium (magnetic susceptibility of Gd^{3+} was taken as 3.2×10^{-4} M^{-1}), μ_0 is the vacuum permeability (1.2566×10^{-6} kg m A^{-2} s^{-2}) and $\boldsymbol{B}$ is the magnetic induction calculated using the Finite Element Method (FEM) (**Figure S2**).

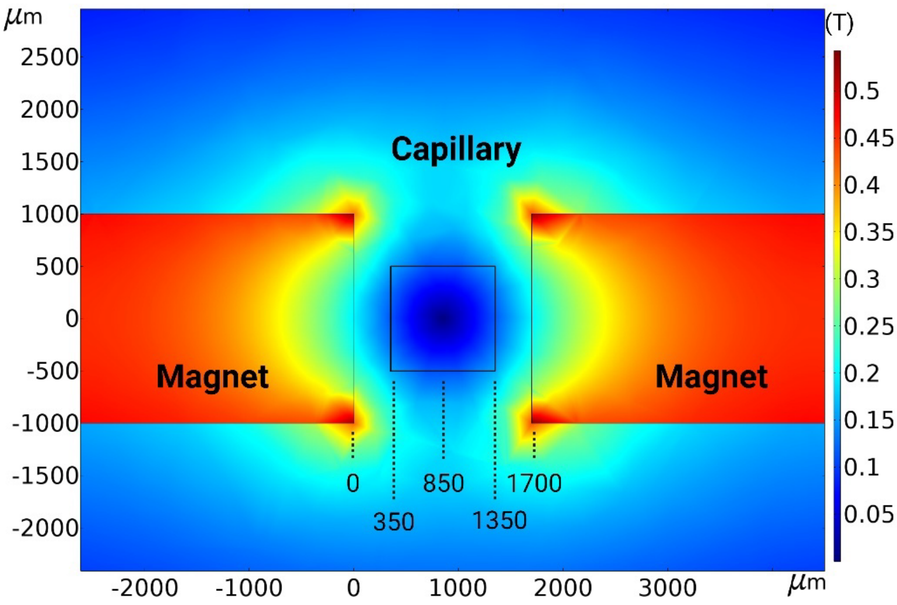

Figure S2. Finite element analysis of magnetic induction ($\boldsymbol{B}$) values inside the capillary.

The drag force acting in the opposite direction to the motion of the microparticle is expressed as the equation shown below:

$$\boldsymbol{F_d} = 6\pi R\eta C_D \mathbf{v} \tag{S4}$$

In this equation, R is the radius of the microparticle, C_D is the drag coefficient (C_D=1 for the microparticle far from the microfluidic channel wall), $\mathbf{v}$ is the velocity of the microparticle, and η is the dynamic viscosity of the medium. Consequently, the force balance equation along the x-direction is rearranged for calculating the viscosity of the solution as follows:

$$\eta = \frac{\frac{4R^2\Delta\chi}{3\mu_0}\left(B_x\frac{\partial B_x}{\partial x} + B_y\frac{\partial B_x}{\partial y} + B_z\frac{\partial B_x}{\partial z}\right)}{6C_D\mathbf{v}} \tag{S5}$$

To conduct the magnetic levitation-based density measurement of the solutions, the platform rotated 90° to orient the opposing magnets parallel to gravity as schematically shown in **Figure 1B**. The microparticles move in the solution with acting magnetic force ($\boldsymbol{F_m}$) and become stationary at an equilibrium height unique to solution density. At this height, where $\boldsymbol{F_m}$ equals the buoyancy force ($\boldsymbol{F_b}$), the force balance equation of the particles is expressed as follows:

$$\boldsymbol{F_m} + \boldsymbol{F_b} = 0 \tag{S6}$$

The buoyancy force is calculated from:

$$\boldsymbol{F_b} = V\Delta\rho\boldsymbol{g} \quad \text{(S7)}$$

, where Δρ (*i.e.*, $\rho_p - \rho_s$) is the volumetric density difference between the microparticle (ρ_p) and the medium (ρ_s) and ($\boldsymbol{g}$) is the gravitational acceleration (9.8 m s^{-2}). In this case, the force balance equation along the z-direction can be rearranged to calculate the density of the solution as follows:

$$\rho_s = \frac{\frac{\Delta\chi}{\mu_0}\left(B_x\frac{\partial B_z}{\partial x} + B_y\frac{\partial B_z}{\partial y} + B_z\frac{\partial B_z}{\partial z}\right)}{\boldsymbol{g}} - \rho_p \quad \text{(S8)}$$

A custom Python code was used to simulate the principle of the MagSity platform. The process involved measuring the viscosity and density of solutions for particles with a size of 15 µm and a density of 1.05 g cm^{-3} as described below.

The microparticle's motion was modeled from the capillary wall (i.e., 1350 µm) to the equilibrium (i.e., 850 µm) for viscosity measurement (**Figure S2**). Initially, **Equation S3** was used to calculate the acting magnetic force ($\boldsymbol{F_m}$). To calculate the positional velocity of the particle, the opposing drag force ($\boldsymbol{F_d}$), which is expressed by **Equation S4**, was equated to the $\boldsymbol{F_m}$ value. The new position was calculated via subtraction of the distance traveled from the initial position at 0.1 s time intervals by taking η as the corresponding viscosity. These steps were repeated for every arrived position to examine the time required to reach the final position. By following these steps, the distinct time intervals between the specified positions (from 1110 to 910 µm which are h_i and h_f, respectively) were calculated for 1-5 cP solution viscosities using different Gd^{3+} concentrations (**Figure 2A**).

The density measurement model was performed by calculating the buoyancy force ($\boldsymbol{F_b}$) exerted on the microparticle employing **Equation S7** and modeling the microparticles' motion, as described previously. The new position was calculated via subtraction of the distance traveled from the capillary wall at 0.1 s time intervals by taking ρ_s as the corresponding medium density. These steps were repeated until $\boldsymbol{F_b}$ equals $\boldsymbol{F_m}$ to reveal the equilibrium levitation height. After these steps, the distinct levitation heights were calculated for 0.95-1.10 g cm^{-3} solution densities using different Gd^{3+} concentrations (**Figure 2B**).

The MagSity platform principle was modeled using a custom-built Python code given below:

```python
from math import pi
import matplotlib.pyplot as plt
import numpy as np
#--------------------------------------------------------------
#Constants
Cd = 1                          # Drag Coefficient
M0 = 1.2566e-6                  # Permeability
initial_position = 1350         # [um]

# Parameters
R = 0.0000075                   # Particle Radius [m]
V = 4/3 * pi * R**3             # Particle Volume [m**3]
TI = 0.1                        # Time Interval [s]

#---# Gadavist Concentrations   # del_X
del_X_200mM = -0.00006418       # Susceptibility Difference
del_X_100mM = -0.00003209
del_X_50mM = -0.00001605

#---# Bdb_x equation y=mx+b     #x=position [um]
m = 0.0522
b = 44.343

#=============================================================#
# Viscosity Model
#=============================================================#

def calculate_force_velocity_distance(del_X, NP, nu):
    Fmag = ((V * del_X) * (m * NP - b)) / M0           # [N]
    velocity = Fmag / (6 * pi * R * Cd * nu)           # [m/s]
    distance = velocity * TI * 1e6                     # [um]
    return Fmag, velocity, distance

def run_simulation(initial_position, final_position, del_X, nu):
    NP = initial_position
    loop_count = 0
```

```
    while NP >= final_position:
        Fmag, velocity, distance =
calculate_force_velocity_distance(del_X, NP, nu)
        NP += distance                                 # [um]
        loop_count += 1

    result_time = loop_count * TI
    return Fmag, NP, result_time
#--------------------------------------------------------------
def print_results(del_X, nu):
    # Gadavist 200mM
    # Initial Position
    initial_position_FP = 1350
    final_position_FP = 1110
    Fmag_FP, final_position_FP, result_time_FP =
run_simulation(initial_position_FP, final_position_FP, del_X, nu)

    print(f"Gadavist {del_X}, Viscosity {nu} mPas")
    print(f"Position {final_position_FP} [um]")
    print(f"Fmag {Fmag_FP} [m/s]")
    print(f"Time to reach: {result_time_FP} [s]\n")

    # Last Position
    initial_position_FL = final_position_FP
    final_position_FL = 910
    Fmag_FL, final_position_FL, result_time_FL =
run_simulation(initial_position_FL, final_position_FL, del_X, nu)

    print(f"Position {final_position_FL} [um]")
    print(f"Fmag {Fmag_FL} [m/s]")
    print(f"Time to reach: {result_time_FL} [s]")
    print(f"Total Time: {result_time_FL - result_time_FP} [s]\n")
#--------------------------------------------------------------
# Test for different viscosity values and susceptibility differences
for nu_value in range(1, 6):            # Test for 1-5 cP
    nu = 0.001 * nu_value               # [mPas]
    print_results(del_X_200mM, nu)
```

```python
    print_results(del_X_100mM, nu)
    print_results(del_X_50mM, nu)

#==============================================================#
# Density Model
#==============================================================#

#Constants
g = 9.8         # [m/s**2]
nu = 0.001      # Solution Viscosity [mPas]
rhop = 1.05     # Particle Density [gcm**-3]
#--------------------------------------------------------------
def calculate_force_velocity_distance2(del_X, EP, rhos, V, m, b, M0,
rhop, R, Cd, nu, g):
    Fmag = ((V * del_X) * (m * EP - b)) / M0          # [N]
    Fb = V * (rhop - rhos) * g * 1000                 # [N]
    velocity = (Fmag - Fb) / (6 * pi * R * Cd * nu)  # [m/s]
    distance = velocity * TI * 1e6                    # [um]
    return Fmag, Fb, velocity, distance

def run_simulation2(initial_position, del_X, nu, rhos):
    EP = initial_position
    loop_count = 0

    while True:
        Fmag, Fb, velocity, distance =
calculate_force_velocity_distance2(del_X, EP, rhos, V, m, b, M0,
rhop, R, Cd, nu, g)

        if abs(Fmag - Fb) < 1e-16:
            break

        EP += distance                                # [um]
        loop_count += 1

    result_time = loop_count * TI
    return Fmag, EP, result_time
```

```
del_X_values = [del_X_200mM, del_X_100mM, del_X_50mM]
rhos_values = [0.9, 0.95,0.96,0.97,0.98,0.99, 1,1.01,1.02,1.03,1.04,
1.05,1.06,1.07,1.08,1.09, 1.1]

def plot_ep_vs_rhos(del_X, label_suffix=""):
    rhos_values = [0.9, 0.95,0.96,0.97,0.98,0.99,
1,1.01,1.02,1.03,1.04, 1.05,1.06,1.07,1.08,1.09, 1.1]

EP_results = []
rhos_results = []

for del_X_value in del_X_values:
    for rhos_value in rhos_values:
        result1, result2, result_time =
run_simulation2(initial_position, del_X_value, nu, rhos_value)

        EP_results.append(result2)
        rhos_results.append(rhos_value)

        print(f"\nResults for del_X = {del_X_value} and rhos =
{rhos_value}:")
        print("Output 1 (Fmag):", result1)
        print("Output 2 (Final EP value):", result2)
#==============================================================#
```

Image Acquisition

A custom-built Python program (Thonny v.4.1.4) was used within the Raspberry Pi mini-computer to regulate the image acquisition parameters, duration and frequency as follows:

```
import time
from time import sleep
import picamera
import datetime as dt
with picamera.PiCamera() as camera:
    camera.start_preview()
   try:
       for i,filename in enumerate(
          camera.capture_continuous('100mM_10Glycerol_Trial1_Viscos
          ity_{counter:d}.jpg')):
          camera.annotate_text=dt.datetime.now().strftime('%H:%M:%S
          ')
          print(filename)
          time.sleep(1)
          if i==600
          break
       finally:
          camera.stop_preview()
```

The captured hologram images are then digitally reconstructed and the reconstructed amplitude images acquired at a distance (z) of 1000 were used to analyze microparticles using the Image J software. For reconstruction, a custom-built Python program was used:

```
import numpy as np
from scipy.fft import fft2, ifft2, fftshift, ifftshift
from matplotlib import pyplot as plt
```

```python
from PIL import Image

image_path =
'C:\\Users\\Öykü\\Desktop\\Analysis\\200mM_30_Viscosity\\t\\200mM_30
Gly_t1_Viscositynp38.jpg'                      # Image file
image = Image.open(image_path)                  # Open Image
object_data = np.array(image.convert('L'))     # Gray Scale

#Parameters
#hologram=hologram input image
#dx,dy=pixel pitch of the object
#n=scaling factor
#_lambda=wave length of the light
#Zmax, Zmin=Maximum and minimum distances of the Fresnel transform
#S=Iteration step size
#d=the distance of the Fresnel Transform
#FileAdress=Reconstructed image saving path

def FNCT(hologram, dx, n, _Lambda, Zmax, Zmin, S, FileAddress):
    dx = 1.1 * 10**-3
    dy = dx

    _Lambda = _Lambda * 10**-6
    Zmax = Zmax * 10**-3
    Zmin = Zmin * 10**-3
    S = S * 10**-3

    num = 1
    count_true = 0

    for ii in np.arange(Zmin, Zmax + S, S):
        k = (2 * np.pi) / _Lambda

        #Hologram Size
        Ny, Nx = hologram.shape
        Nxx = Nx
        Nyy = Ny
```

```
        #Spatial Domain
        x = np.ones((Nyy, 1)) * np.arange(-Nxx/2, Nxx/2) * dx
        y = np.arange(-Nyy/2, Nyy/2)[:, np.newaxis] * np.ones((1,
Nxx)) * dy

        Lx = dx * Nxx
        Ly = dy * Nyy
        dfx = 1 / Lx
        dfy = 1 / Ly

        #Frequency Domain
        u = np.ones((Nyy, 1)) * np.arange(-Nxx/2, Nxx/2) * dfx
        v = np.arange(-Nyy/2, Nyy/2)[:, np.newaxis] * np.ones((1,
Nxx)) * dfy

        if np.any(np.sqrt(u**2 + v**2) < (n / _Lambda), axis=None):
            print('True')
            count_true += 1

        #Fourier Transform of the Hologram (O)
        O = fft2(fftshift(hologram))

        #Transfer Function (H)
        H = np.exp((1j * k * ii * n) * np.sqrt(1 - (_Lambda * (u /
n))**2 - (_Lambda * (v / n))**2))
        H = fftshift(H)

        #Reconstructed Hologram (o)
        o = ifftshift(ifft2(O * H))

        du = dx
        dv = dy

        #Image Processing

        #Real Component
```

```
    d = np.real(o)
    d = (d - np.min(d)) / np.ptp(d)
    realcomp = d

    #Imaginary Component
    d = np.imag(o)
    d = (d - np.min(d)) / np.ptp(d)
    imagcomp = d
    re = np.real(o)
    re = (re - np.min(re)) / np.ptp(re)

    #Phase Component
    img = np.imag(o)
    img = (img - np.min(img)) / np.ptp(img)
    phase = np.arctan2(img, re)
    d = np.abs(o)
    d = (d - np.min(d)) / np.ptp(d)
    ampp = d
    d = np.angle(o)
    d = (d - np.min(d)) / np.ptp(d)
    phasecomp = d
    value = int(ii * 10**3)

    filename = f"amplitude_{num}_{int(ii * 10**3)}.jpg"
    plt.imsave(FileAddress + '/' + filename, ampp, cmap='gray')

    num += 1

print(f"{count_true} amplitude images created.")
```

Materials and Solution Preparation

Polyethylene microspheres having 1.05 g cm^{-3} densities and 15 µm diameter (microParticles GmbH, Germany) and Gadavist® (Gd^{3+}, Bayer, Germany) as a paramagnetic medium were obtained to be utilized in experiments. Glycerol and Bovine Serum Albumin (BSA) were bought from Sigma Aldrich, (Germany). Dulbecco's modified Eagle's high glucose (DMEM), Roswell Park Memorial Institute (RPMI), phosphate-buffered saline (PBS) and fetal bovine serum (FBS) were purchased from Thermo Fisher Scientific (USA).

Samples were prepared according to established protocols. FBS-DMEM solutions containing 5%, 10%, 20%, and 30% FBS were prepared by diluting DMEM with the corresponding volume of FBS. Bovine serum albumin (BSA) solutions were prepared by dissolving BSA powder in either FBS-DMEM or PBS to achieve final concentrations of 0.5 mM, 1 mM, and 2 mM. Spent cell culture medium was collected from HUVEC (Human Umbilical Vein Endothelial Cells), human breast cancer cell lines of MCF-7 (Michigan Cancer Foundation-7) and MDA-MB-231.

MagSity Platform Calibration

To model the viscosity range of clinical samples with a viscosity of 1-5 cP,[2] glycerol-water solutions are prepared accordingly as weight percentages. The viscosity of the prepared solutions is measured with a commercial viscometer (Brookfield DV-II+ Pro) and density is calculated by measuring the mass of the solution via an analytical balance (AND GH-252) and subsequently dividing it by its volume (**Figure S3** and **S4**). Polyethylene standard microparticles of 15 µm diameter and density of 1.05 g cm^{-3} are used as microsensors in the platform. A total volume of 50 µL specimen is prepared by adding microparticles with a final concentration of 10^5 particles mL^{-1} into the sample of interest in the presence of Gd^{3+}. The capillary is loaded with 30 µL of the specimen and placed between the opposing magnets after being sealed. The calibration of the MagSity platform was performed for both viscosity and density measurements. For this purpose, prepared microparticles were levitated in different concentrations of Glycerol solutions containing 50-200 mM Gd^{3+}. The experiments conducted at 26 and 37 °C were carried out in an incubator (NB-T205, N-BIOTEK).

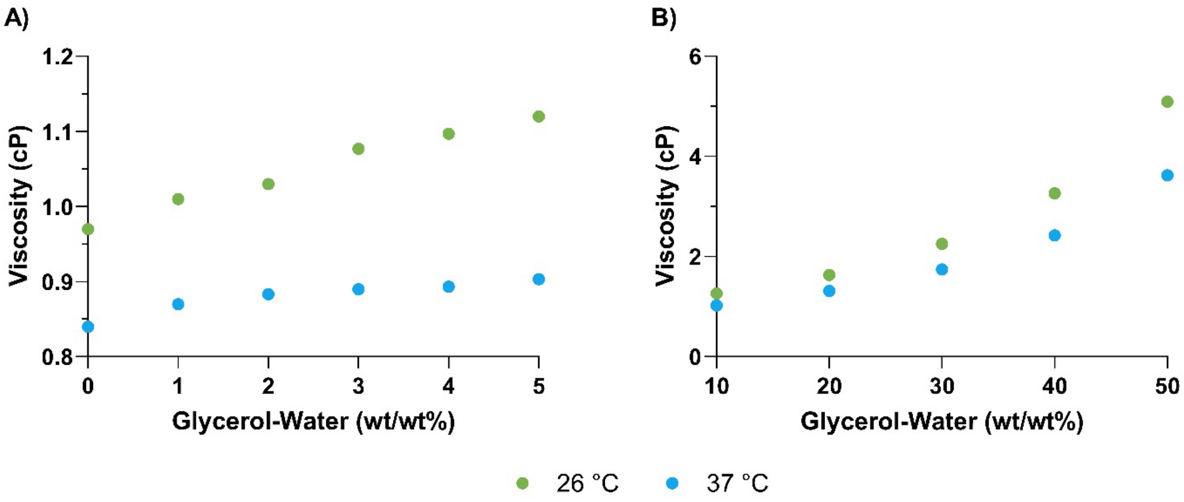

Figure S3. Measured viscosity values of A) 0-5% and B) 10-50% Glycerol solutions at 26 °C and 37 °C.

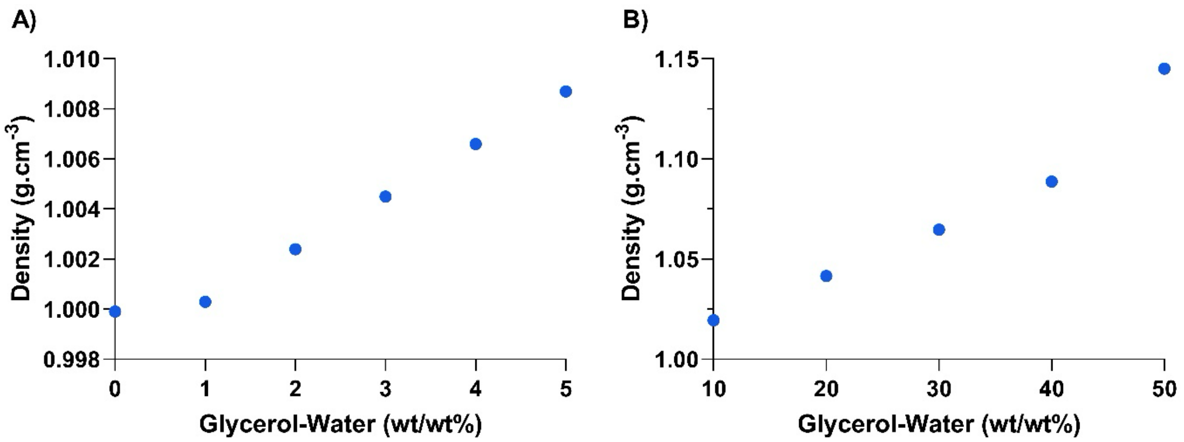

Figure S4. Measured density values of A) 0-5% and B) 10-50% Glycerol solutions at 26 °C.

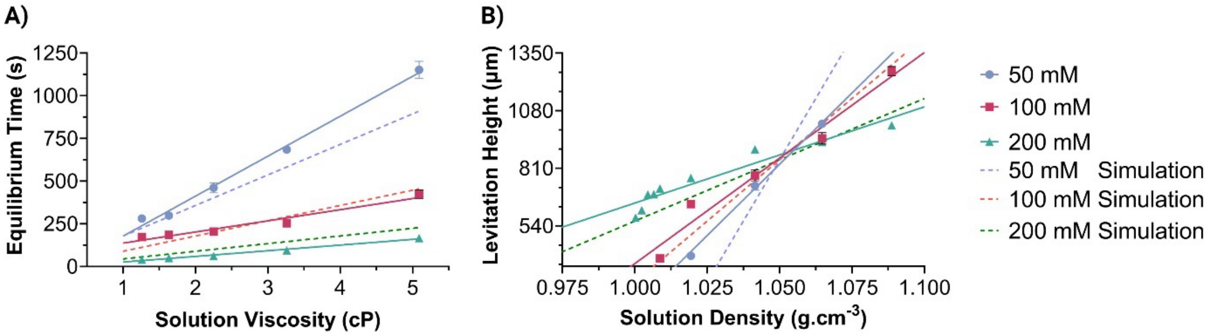

Figure S5. Comparison of experimental and simulated results for A) viscosity and B) density measurements. Dashed lines depict simulation results, while solid lines represent curve fittings to the experimental data.

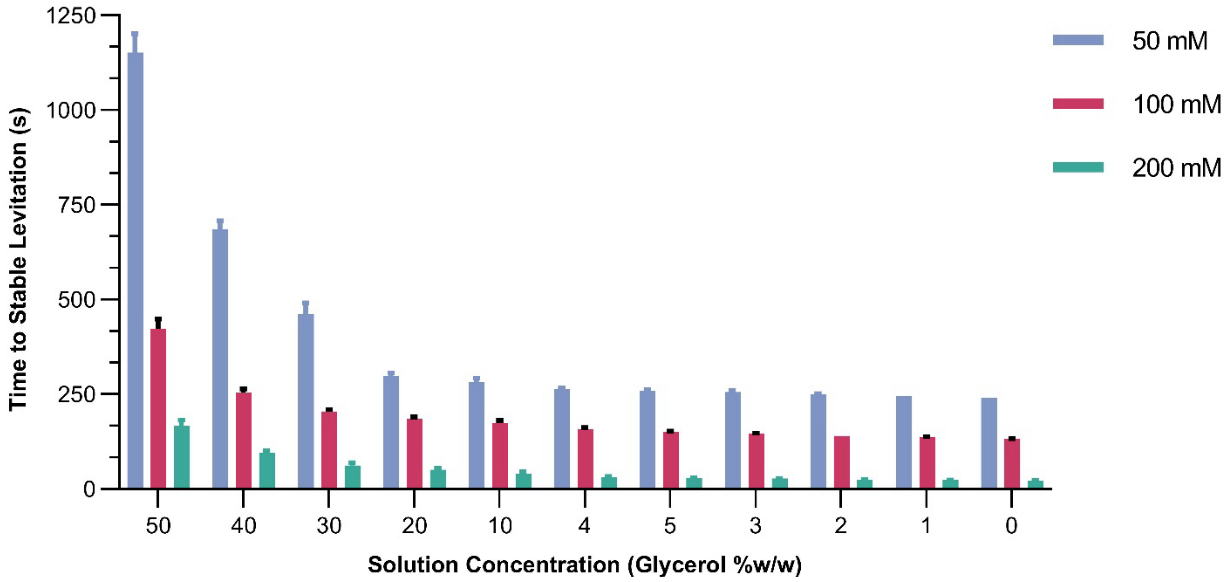

Figure S6. Time to reach stable levitation height for different Glycerol solutions using 50-200 mM Gd^{3+}.

Cell Culture Protocol

Cells (HUVEC, MCF-7 and MDA-MB-231) were cultured in DMEM medium supplemented with 10% FBS and 1% penicillin/streptomycin at 37 °C in a humidified environment with 5% CO_2. Upon reaching confluency, cells were passaged per standard protocols with an initial seeding density of 1×10^6 cells. For experimental purposes, cells grew undisturbed for 7 days without passaging to ensure sufficient accumulation of secreted factors in the spent medium. Following the incubation period, the spent medium was carefully collected and then used in the experiments.

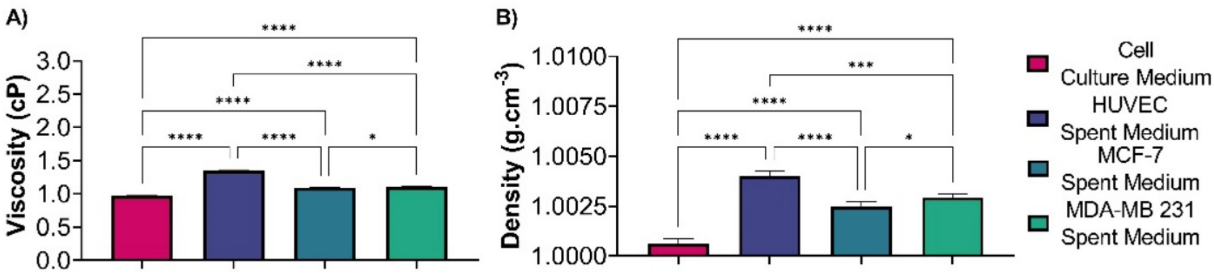

Figure S7. Comparison of A) viscosity and B) density measurements between fresh cell culture medium (i.e., DMEM supplemented with 5% (v/v) FBS) and spent mediums obtained from 7-day cultures of HUVEC, MCF-7, and MDA-MB-231 cell lines. Measurements were conducted using the MagSity platform with 100 mM Gd^{3+}. *, *** and **** show $P\leq0.05$, $P\leq0.001$ and $P\leq0.0001$, respectively.

Table S1. Performance of the MagSity platform. Solutions with different densities (1.00-1.089 g cm^{-3}) and viscosities (1.26-5.09 cP) were measured in the platform using 50-200 mM Gd^{3+} at 26 °C.

Density Measurement		
Gd^{3+}	Sensitivity (μm/g cm^{-3})	Analysis Time (min)
50 mM	7.49×10^{-5}	5.79 ± 1.38
100 mM	1.01×10^{-4}	3.25 ± 0.56
200 mM	2.22×10^{-4}	0.67 ± 0.37
Viscosity Measurement		
Gd^{3+}	Sensitivity (s/cP)	Analysis Time (min)
50 mM	4.27×10^{-3}	6.66 ± 4.52
100 mM	1.53×10^{-2}	3.19 ± 1.36
200 mM	3.01×10^{-2}	0.86 ± 0.70

Table S2. Comparison of the MagSity platform with the state-of-the-art miniaturized devices for viscosity and density measurements.

Sensing Mechanism	Sample Volume (μL)	η Range (cP)	η Measurement Accuracy (%)	ρ Range (g cm^{-3})	ρ Measurement Accuracy (%)	Measurement Time (min)	Measurement Method	Specimen	Device Cost ($)	Reference
Microfluidic Pressure Sensing (RheoSense®)	20	0.2-100,000	98	NA[a)]	NA	<10	Measurement of the pressure drop across a microchannel via pressure sensors.	Various solutions	>10,000	[3]
Microfluidic Particle Tracking Velocimetry	<100	1-1000	NR[b)]	NA	NA	<20	Digital Holographic Microscope for particle tracking velocimetry	Water/Polyethylene oxide solutions	NR	[4]
Microfluidic Flow Sensing	~10	0.7-10	96	NA	NA	NR	Measurement of flow rate across a microchannel via flowmeters.	Antibody solutions	NR	[5]
Microcapillary	~5	1-600	NR	NA	NA	<10	Monitoring capillary pressure-driven fluid mean velocity.	Various polymer solutions	NR	[6]
Microsensor Based	80-400	5-100	>89.7	NA	NA	NR	The flow-induced deflection of PDMS micro-pillars is analyzed using machine learning algorithms.	Glycerol/Water solutions	NR	[7]
Microfluidic Flow Comparator	1500	1-1000	NR	NA	NA	>10	Monitoring the co-flow interface mismatch of a known viscosity reference fluid and test fluid.	Glycerol/water solutions, polymeric fluids	NR	[8]
Droplet-Based	NR	20-70	NR	NA	NA	NR	Monitoring droplet behavior using high-speed microscopy	Glycerol/Water solutions	NR	[9]
MEMS microcantilever	<150	0.86-3.02	NR	NA	NA	NR	Optical detection of phase and frequency by an interferometric readout of a magnetically actuated microcantilever	Glycerol/Water solutions, FBS, BSA, Blood Plasma	NR	[10]

MEMS Magnetoelastic Sensor	100	1.0-3.0	NR	NA	NA	NR	Detection of resonant frequency of a magnetically actuated magnetoelastic sensor by a planar coil.	Glycerol/Water, Blood Plasma	NR	[11]
MEMS Piezoelectric Sensor	NR	0.39-7.36	NR	0.68-0.9	NR	NR	Electrical measurement of natural frequency and quality factor	Silicone oils, Isopropanol, Heptane solutions	NR	[12]
MEMS Microcantilever	100	0.935-4	NR	0.995-1.15	NR	NR	Monitoring the oscillation frequency shifts by laser doppler vibrometer of two geometry cantilevers	Ethylene/Glycerol/Water solutions	NR	[13]
MagSity	30	0.84-5.09	>97.7	1.00-1.09	>99.9	<7	Digital holographic microscope to determine particle levitation velocity and height	Glycerol/Water, various biological solutions	<120	This study

[a)]Not Applicable; [b)]Not Reported

Table S3. Comparison of the MagSity platform with the state-of-the-art MagLev platforms.

Measured Properties			Platform Properties		Reference
Solution Viscosity	Solution Density	Particle Density	Integrated Imaging System	Novelty	
-	Droplet-based	+	+	High-throughput MagLev-based particle and droplet solution density measurement	[14]
-	−	+	−	Density-based exosome detection using MagLev	[15]
-	Droplet-based	+	−	MagLev-based particle and water-immiscible droplet density measurement	[16]
-	−	+	−	High-sensitivity MagLev-based particle density measurement	[17]
-	−	+	+	Novel integration of MagLev with lensless holography for particle density measurement	[18]
+	+	-*	+	New methodology using known particles to determine the density and viscosity of unknown solutions	This study

* While it can be used for particle density measurement, it is not the focus of this study.

Table S4. Cost table of the MagSity platform and single analysis.

Magsity Platform	
Component	**Cost ($)**
LED	1.6
Pinhole	60
Camera Module v2	20
Raspberry Pi 3B+	35
3D Printed Platform Support	0.5
N52 Magnet (2 pieces)	1.6
Total	118.7
Single Test	
Component	**Cost ($)**
Capillary (1 piece)	2.27
Gadavist (0.5 μL)	0.01
Microparticles (2 μL)	0.08
Total	2.36

Video S1. Video of experimental procedure conducted on the MagSity platform including sample preparation, setup assembly and measurement principle.

References

[1] J. Zhang, W. Li, G. Alici, in *Advanced Mechatronics and MEMS Devices II* (Eds: D. Zhang, B. Wei), Springer International Publishing, Cham, **2017**, pp. 563–593.
[2] M. Mohammadi Aria, A. Erten, O. Yalcin, *Front. Bioeng. Biotechnol.* **2019**, *7*, 395.
[3] "RheoSense Viscometers," can be found under https://www.rheosense.com/products/viscometers, **n.d.**
[4] S. Gupta, S. A. Vanapalli, *Physics of Fluids* **2020**, *32*, 012006.
[5] S. D. Hudson, P. Sarangapani, J. A. Pathak, K. B. Migler, *Journal of Pharmaceutical Sciences* **2015**, *104*, 678.
[6] N. Srivastava, M. A. Burns, *Anal. Chem.* **2006**, *78*, 1690.
[7] A. Mustafa, D. Haider, A. Barua, M. Tanyeri, A. Erten, O. Yalcin, *Sensors & Diagnostics* **2023**, *2*, 1509.
[8] D. E. Solomon, S. A. Vanapalli, *Microfluid Nanofluid* **2014**, *16*, 677.
[9] E. André, N. Pannacci, C. Dalmazzone, A. Colin, *Soft Matter* **2019**, *15*, 504.
[10] O. Cakmak, C. Elbuken, E. Ermek, A. Mostafazadeh, I. Baris, B. Erdem Alaca, I. H. Kavakli, H. Urey, *Methods* **2013**, *63*, 225.
[11] P. Chen, Q. Jiang, S. Horikawa, S. Li, *J. Electrochem. Soc.* **2017**, *164*, B247.
[12] T. Manzaneque, V. Ruiz-Díez, J. Hernando-García, E. Wistrela, M. Kucera, U. Schmid, J. L. Sánchez-Rojas, *Sensors and Actuators A: Physical* **2014**, *220*, 305.
[13] O. Cakmak, E. Ermek, N. Kilinc, G. G. Yaralioglu, H. Urey, *Sensors and Actuators A: Physical* **2015**, *232*, DOI 10.1016/j.sna.2015.05.024.
[14] S. Ge, Y. Wang, N. J. Deshler, D. J. Preston, G. M. Whitesides, *J Am Chem Soc* **2018**, *140*, 7510.
[15] A. B. Sözmen, A. Arslan-Yildiz, *ACS Sens.* **2024**, *9*, 2043.
[16] K. A. Mirica, S. S. Shevkoplyas, S. T. Phillips, M. Gupta, G. M. Whitesides, *J. Am. Chem. Soc.* **2009**, *131*, 10049.
[17] A. Nemiroski, A. A. Kumar, S. Soh, D. V. Harburg, H.-D. Yu, G. M. Whitesides, *Anal. Chem.* **2016**, *88*, 2666.
[18] K. Delikoyun, S. Yaman, E. Yilmaz, O. Sarigil, M. Anil-Inevi, K. Telli, O. Yalcin-Ozuysal, E. Ozcivici, H. C. Tekin, *ACS Sens.* **2021**, *6*, 2191.