Supplementary Information

Micro-a-fluidics ELISA for Rapid CD4 Cell Count at the Point-of-Care

ShuQi Wang [a], Savas Tasoglu [a], Paul Z. Chen [a], Michael Chen [a], Ragip Akbas [b], Sonya Wach [a], Cenk Ibrahim Ozdemir [a], Umut Atakan Gurkan [a], Francoise F. Giguel [c], Daniel R. Kuritzkes [d], Utkan Demirci [a, d, e, *]

[a] *Bio-Acoustic-MEMS in Medicine (BAMM) Laboratory, Division of Biomedical Engineering, Department of Medicine, Brigham and Women's Hospital, Harvard Medical School, Boston, MA 02139, USA*

[b] *Özyeğin University, Civil Engineering Department, Istanbul, Turkey.*

[c] *Infectious Diseases Unit, Massachusetts General Hospital, Boston, MA 02114, USA.*

[d] *Division of Infectious Diseases, Brigham and Women's Hospital, Harvard Medical School, Boston, Massachusetts 02115, USA*

[e] *Harvard-MIT Health Sciences and Technology, Cambridge, MA 02139, USA.*

[*] *Corresponding author:*

Utkan Demirci, Ph.D.

Bio-Acoustic-MEMS in Medicine (BAMM) Laboratory, Harvard-MIT Health Sciences and Technology

65 Landsdowne St., # 267

Cambridge, MA 02139, USA

udemirci@rics.bwh.harvard.edu

Construction of micro-a-fluidic instrument

To automate the micro-a-fluidic CD4 cell count assay, a small instrument with a built-in motorized stage was constructed (**Fig. S1**). The instrument was designed to have three layers for easy assembly. The base was a sheet of acrylic measuring 7" x 12" with a thickness of 0.25". A motorized stage was placed on the center on the base sheet. Another wall (0.5" tall and 0.125" thick) was constructed 0.125" away from the edges of the sheet, in order to allow the second layer to attach to the base. The second layer was designed to hold m-ELISA chips and to allow small magnets to move under each chip. The second layer was composed of three separately designed sub-layers, which was permanently attached together. This layer had a 3 x 3 grid of rectangular slots to allow the micro-a-fluidic chips to be placed inside. The top layer was designed to allow a camera or cellphone to capture images of each chip at uniform luminosity. To fully automate the movement of magnetic beads, a computer program was written in C to allow for bidirectional communication with the stage, thus realizing automated CD4 cell count. Execution commands were sent to the stage to control the speed and acceleration of movement of the motorized stage. Thus, the conditions of m-ELISA were optimized by changing the program with respect to the acceleration, speed and time of movement of magnetic beads in the micro-chambers.

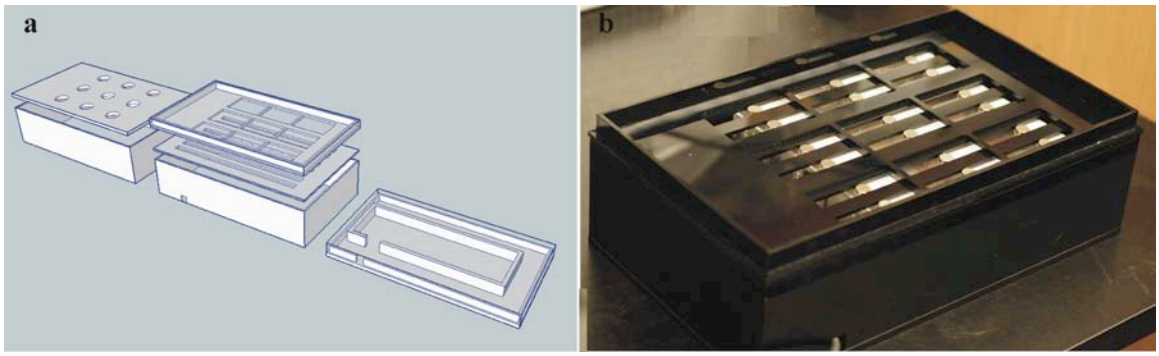

Fig. S1 Schematic of micro-a-fluidic instrument. (**a**) Schematic design. (**b**) Image of micro-a-fluidic instrument built for CD4 cell count.

For the control of the instrument, the stage was connected to a computer via RS-232 serial port converted to USB. The stage was also connected to an AC adapter for additional power. Basic execution of commands was performed by first polling the

device until a 'ready' signal was received, and then sending the command. Commands are given as short character strings, such as "M:1+P5000", to send a message to move the stage in the forward direction by 5000 pulses. The speed and acceleration of movement could also be specified via commands, allowing for a large amount of control over the movement of the stage. Using such commands, the stage and magnets were moved back and forth to manipulate the magnetic beads in the micro-a-fluidic chambers for use in m-ELISA. The program automates the entire testing process.

Table S1 Comparison of m-ELISA with other existing ELISA methods

	Conventional ELISA	**Microfluidic ELISA**	**m-ELISA**
Assay time	4-6 hours	20 min- 6 hours	10 min
Labor intensive	Yes	Likely, depending on the setup.	No
Automation potential	Maybe	Likely, depending on the setup.	Automated with a simple setup
Point-of-care testing potential	No	Likely, depending on the setup.	Yes

Modeling of magnetic forces in a micro-a-fluidic device

To calculate magnetic forces on magnetic beads, we solved Maxwell equations using the finite-difference method. The computational model included a magnetic bead in a liquid chamber and a cylindrical magnet. In this model, the surrounding medium was set sufficiently large such that the magnetic field was tangential along the boundaries. This condition was satisfied by the requirement of closed loops around the magnet ($n \cdot B = 0$), where n is unit vector normal to the surface, and B is the magnetic flux density. As a constitutive equation for the fluidic chamber and the surrounding air, $B = \mu_0 \mu_r H$ was employed. Here, H is the magnetic field density, μ_0 is the permeability of free space ($4\pi \cdot 10^{-7} \mathrm{N} \cdot \mathrm{A}^{-2}$), and μ_r is the relative permeability coefficient. At some critical height (h) above the surface of magnets, magnetic forces are not large enough to overcome the Brownian motion. To predict the limitation of our design, we assessed this height as a function of the magnetic bead size. Critical height (h) for magnetic particles is calculated by the following equation.

$$F_z(z = h) = \mu_0 V_p f(H_a) \frac{{M_s}^2 {R_m}^2}{4} \left[\left(\frac{h}{\sqrt{h^2 + {R_m}^2}} - \frac{h + L_m}{\sqrt{(h + L_m)^2 + {R_m}^2}} \right) \left(\frac{1}{(h^2 + {R_m}^2)^{3/2}} - \frac{1}{((h + L_m)^2 + {R_m}^2)^{3/2}} \right) \right] = \frac{kT}{2R_p} \quad (1)$$

$$H_a(z) = \frac{M_s}{2} \left[\frac{z + L_m}{\sqrt{(z + L_m)^2 + {R_m}^2}} - \frac{z}{\sqrt{z^2 + {R_m}^2}} \right] \quad (2)$$

$$f(H_a) = \begin{cases} \dfrac{3(\chi_p - \chi_f)}{(\chi_p - \chi_f) + 3} & H_a < \left(\dfrac{(\chi_p - \chi_f) + 3}{3\chi_p} \right) M_{sp} \\ M_{sp}/H_a & H_a \geq \left(\dfrac{(\chi_p - \chi_f) + 3}{3\chi_p} \right) M_{sp} \end{cases} \quad (3)$$

where $M_{sp}(= 6 \text{ x } 10^5$ A/m) and $M_s(= 1.14 \text{ x } 10^6$ A/m) are saturation magnetization of magnetic particles and magnet, respectively (**Table S2)**. Here, R_m $(= 3/8$ inch) and $L_m(= 1$ inch) are the radius and length of magnet. V_p, μ_0, H_a are volume of the magnetic beads, free-space magnetic permeability $(= 1.257\ x\ 10^{-6}\ N/A^2)$, the applied magnetic field intensity at the center of magnetic beads, respectively. χ_p and χ_f are the magnetic susceptibilities of the particle and fluid, respectively. A magnetization model is adopted that is consistent with the magnetic beads we used, $\chi_p \gg 1$. Then Eq. 3 simplifies to Eq. 4 as shown below.

$$f(H_a) = \begin{cases} 3 & H_a < M_{sp}/3 \\ M_{sp} & H_a \geq M_{sp}/3 \end{cases} \quad (4)$$

Table S2 Parameters for the computational and mathematical model.

Parameter	Definition	Value
μ_0	Permeability of free space	$4\pi \cdot 10^{-7}$ N/A^2
M_{sp}	Saturation magnetization of magnetic particles	$6 \text{ x } 10^5$ A/m
M_s	Saturation magnetization of magnetic particles and magnet	$1.14 \text{ x } 10^6$ A/m
R_p	Radius of the magnetic beads	1 μm
R_m	Radius of magnet	$3/8$ inch
L_m	Length of magnet	1 inch
γ	Surface tension of mineral oil against PBS	50 mN/m

Magnetic simulation results showed that at the edge of the circular permanent magnet, both vertically and horizontally pulling magnetic forces peaked due to a fringe effect (**Fig. 2e**). Drag force exerted on the magnetic beads was approximated with Stokes' drag equation due the small Reynolds number of our system, $F_d(\dot{x}) = 6\pi\mu R_p \frac{dx}{dt}$, where μ is viscosity of the fluid in the chamber and R is radius of the magnetic beads. As viscosity of the medium and/or velocity of the bead increased, drag forces also increased (**Fig. S2b**).

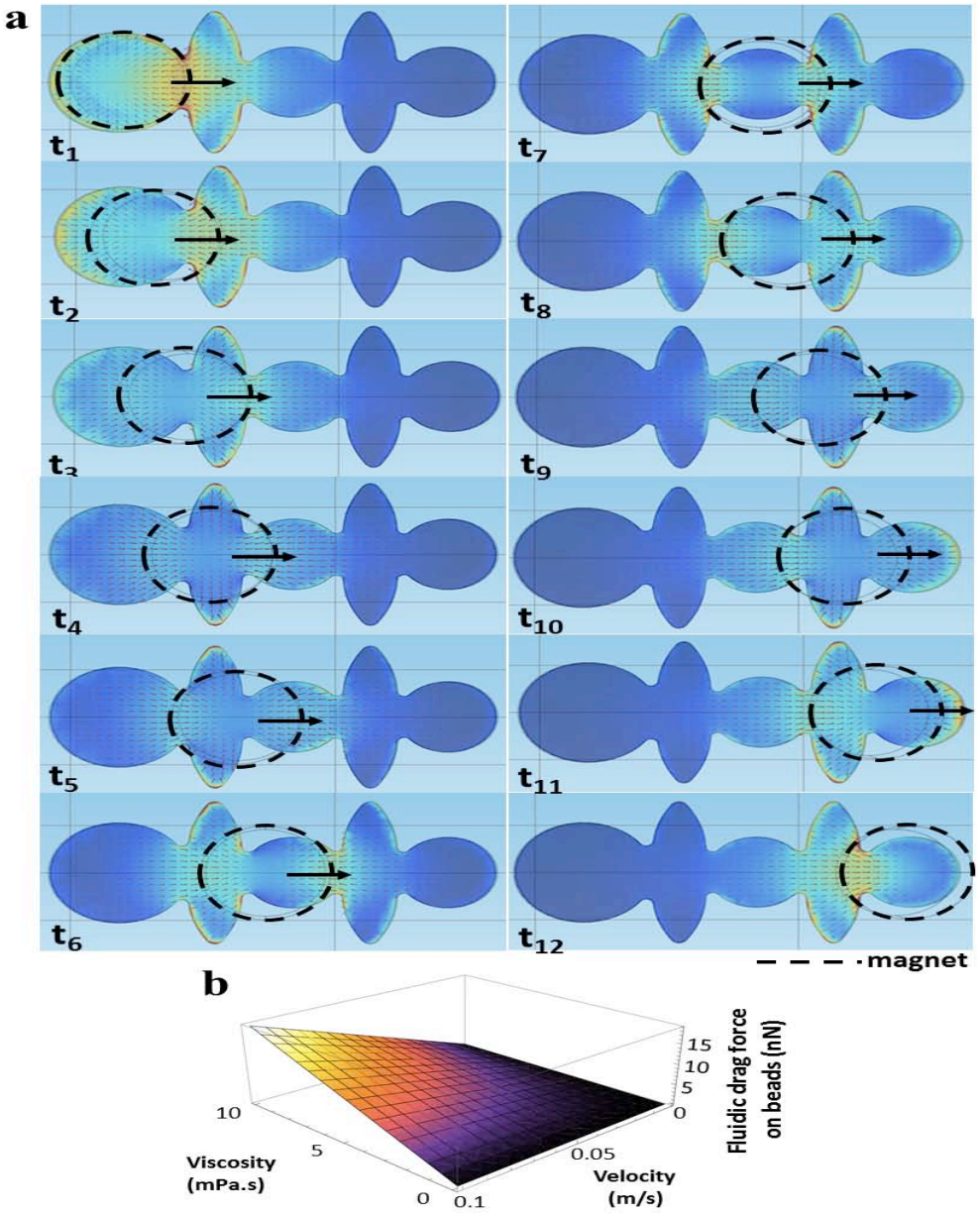

Fig. S2 (a) Simulation results of magnetic flux density (contours) and unit magnetic force along the horizontal plane (arrows) when a magnet moves horizontally from left to right. Results are plotted for a range of time points (top-to-bottom, left-to-right). Dashed lines are the circular edges of the cylindrical magnet (top view). **(b)** Fluidic drag forces on a single magnetic bead as a function of bead velocity and viscosity of the surrounding medium. Radius of beads is set to 1 micron. Viscosity range was chosen based on the materials used (PBS and oil). Velocity range was chosen large enough to include bead velocities observed in the experiments.

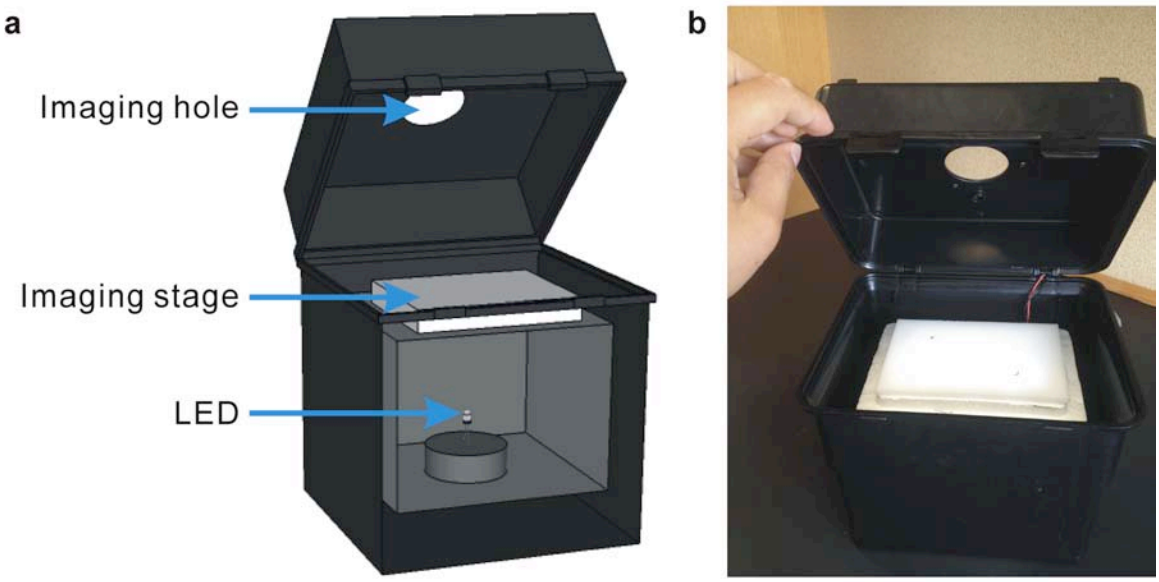

Fig. S3 (**a**) Schematic design. (**b**) Setup of image-taking to avoid interference of external conditions (*e.g.* lighting).

Mobile and desktop applications for CD4 cell count

The software applications were designed to detect or pick image regions that represent the color intensity within the microchannels, to average pixel values therein (red channel) and to report $CD4^+$ T lymphocyte cell count. The mobile and desktop versions of applications provide the same analysis functionality and give equivalent results. The primary difference is that the desktop application allows batch processing for multiple images in the same folder. The desktop application works on Windows operating systems and the mobile application was written for Windows Phone 7 and 8 operating systems. In addition, the cell phone code and algorithms are applicable to various other smart phone platforms. The mobile application can process images captured by a smart phone camera as well as previously saved photos.

To facilitate region selection by the application, four vertically aligned markers were placed on every microchip (**Fig. S4a**). The application automatically detects marker candidates to infer image regions for analysis using the flood-fill algorithm. This algorithm obtains continuous regions of pixels within specific RGB value range and pixel quantity. These value ranges and pixel quantities were based on the clinical images obtained, and were implemented as user-defined modifiable parameters. Out of marker candidates calculated by flood-fill algorithm calculated for each image, a search is performed to estimate four valid vertically aligned markers. The application assumes that the captured microchip images were oriented horizontally with some allowable rotation angles. Once the markers are detected, regions that correspond to two circular microchip regions and background regions are calculated. The desktop and mobile applications also allow the user to manually pick and update regions.

Following the selection of the image region, as indicated by the red circle, the desktop application was designed to convert the color intensity to RGB pixel values (**Fig. S4b**). The application discards pixels within regions that have very low RGB values as invalid values. To minimize the illumination difference from chip-to-chip, we normalized the color intensity based on the difference in the background between the standard and sample images. The application selects a typical background region from each sample

image and compares the R values therein to an average of R values from the background regions in standard images. The R values from the selected region in sample images were offset by deducing the R value difference. After normalization, the pixel values exported from standard images were used to generate the standard curve, by which the number of $CD4^+$ cells in clinical samples was calculated and reported.

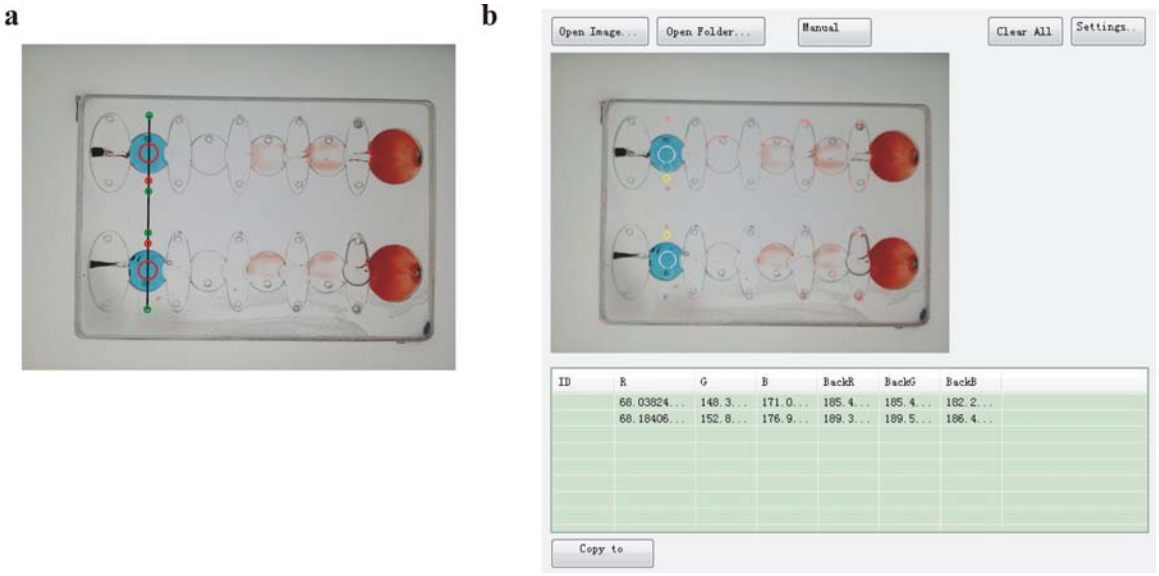

Fig. S4 (a) Schematic representation of automated region detection. The application detects the four markers on the chip and estimates the location of colorimetric regions and backgrounds using the markers. **(b)** A sample screenshot for automatically detected regions on the desktop application. White and yellow circles represent colorimetric regions and background regions respectively. Red colored areas represent candidate markers.

The mobile application runs the same way the desktop application. In addition, it allows user to enter standard curve values, calculates standard curve values from images, and draws chart for standard curve. The application reported R values obtained from the automatically selected regions within the microchannels and the number of $CD4^+$ T lymphocytes (cells/μL) on the cell phone screen (**Fig. S5a**). **Fig. S5b** shows an example standard curve stored in the program.

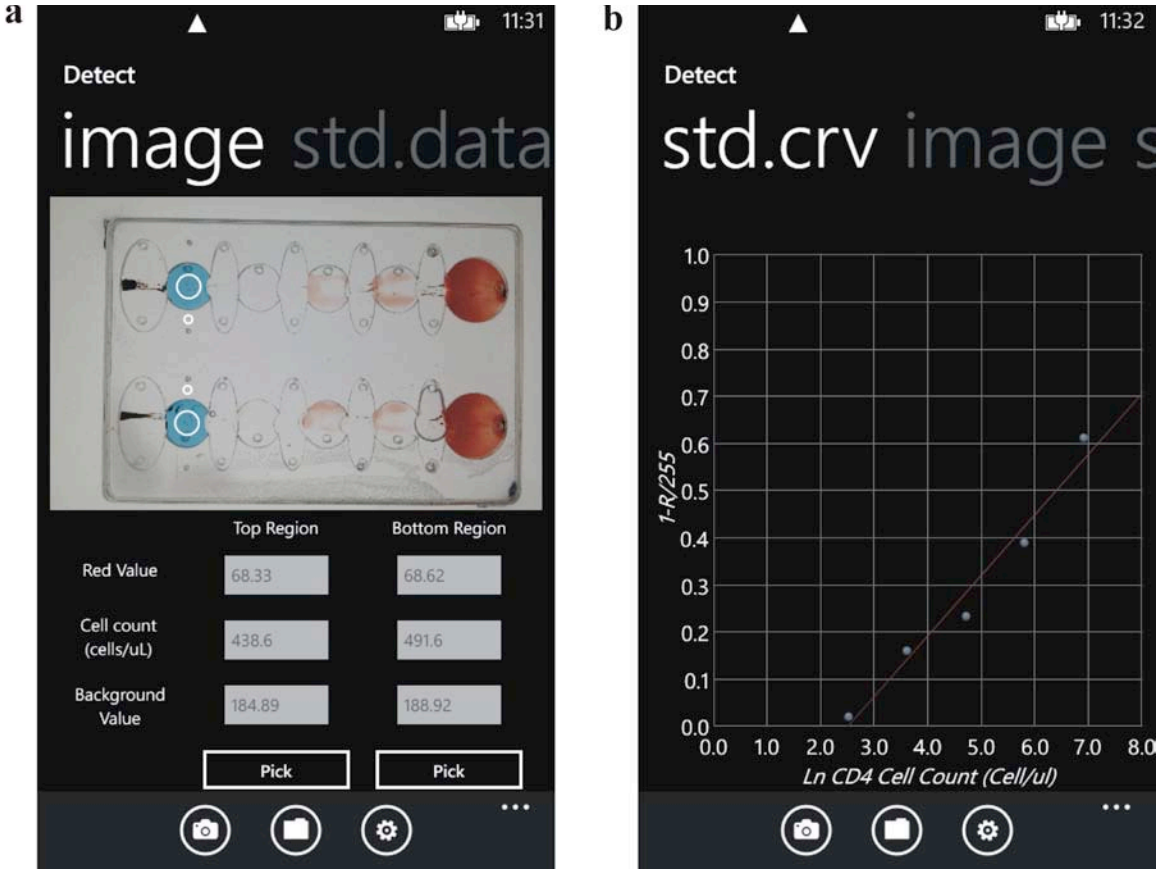

Fig. S5 Screen shots of Mobile application for CD4 cell count running on Windows Phone 8. **(a)** Screenshot of CD4 cell count program when reporting the result. **(b)** Screenshot of an example standard curve of micro-a-fluidic ELISA. Used with permission from Microsoft.

Mobile Application Instructions

The desktop application runs on any Windows PC. To execute the mobile application, a phone with Windows Phone 7 or Windows Phone 8 operating system can be used. To install the application, 'MarkerDetect.xap' was deployed using 'Application Deployment' tool that comes with Windows Phone Developer tools.

The main page of the mobile application allows users to choose and analyze an image. Users can choose an image by taking a photo using the mobile camera or selecting an image loaded onto the cell phone. To calculate the standard curve using standard images, click settings and check 'standard curve calibration'. Standard images must be loaded in order starting from higher CD4 cell counts to lower ones for regions to be assigned to a certain CD4 cell count. After loading the images, the mobile application calculates R values and updates standard curve regression parameters. Users can swipe the screen to edit standard curve values and show standard curve graph. To analyze the clinical testing images, click settings and uncheck 'standard curve calibration'. The application will detect regions, report R values, background values, and CD4 cell count for each selected image. Users can zoom into the image and manually update selected regions within the image.

The source code for the application was included in the supplement and the key source files are described (**Table S3**).

Table S**3** Key source files for desktop and mobile application

Desktop App File	**Mobile App File**	**Purpose**
MarkerDetectWin.sln	MarkerDetect.sln	Visual Studio solution file for the application
Program.cs	App.xaml	Entry point for the application
AppData.cs	AppSettings.cs	Storage location for application data
MainFrm.cs	MainPage.xaml	Primary code for the UI behavior
Settings.cs	SettingsPage.xaml	Settings UI
FloodFill.cs	FloodFill.cs	Flood Fill algorithm implementation
FloodFillQueue.cs	FloodFillQueue.cs	
ChipImageAfluidic.cs	ChipImageAfluidic.cs	Image-related code