What is LightSensorArray

LightSensorArray is an array of 8 sensors to detect color equivalent gray scales in front of the sensor window. It has a built in light source for each of its sensors and can detect and differentiate dark and light shades.

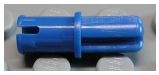
Connections and Placement

LightSensorArray can be connected to any of the four sensor ports of NXT or EV3 by using standard cables from NXT set, or FlexiCable from mindsensors.com.
While attaching sensor to your robot, face the long clear sensor window towards the mat, placing it approx 5 millimeters above the mat surface (approx $\frac{1}{4}$ inch).
If you plan to use this sensor to track a line, place the sensor in front of the driving wheels – (in the direction the robot will be moving).

Mounting LightSensorArray on your contraption

The holes on the LightSensorArray enclosure are designed for tight fit of Technic pins (or axles) with '✚' cross section. The holes however are not designed for repeated insertions/removals of these pins.

To mount LightSensorArray on your contraption we suggest that you use two dark gray 'Technic Axle 3 with Stud' as shown.
Insert axles from the top of the LightSensorArray and secure with a bushing on the back or mount it on your contraption directly.
Alternately, you may use blue 'Technic Axle Pin with Friction', as shown.



While disassembling contraption, leave the pins on LightSensorArray.

Sensor Calibrations

⚠ **NOTE**

For best results, LightSensorArray should be calibrated for the surface it will be used on. To calibrate the sensor, keep the sensor window on white area, and calibrate white. Then keep the sensor window on black area and calibrate black. While placing on the white or black area, ensure that all sensors are over the same color surface.

Calibration programs can be found in the sample programs of the respective

programming environment. Download sample programs from the Programming Techniques section below.

Sensor Operations

You can use the raw sensor readings of the sensor and compute yourself whatever you need from it. The raw values are returned as byte per sensor with 0 indicating black and 100 indicating white.

⚠ **NOTE**

In order to write a good line follower program, you will need to know and understand the **'PID Control Theory'**. http://www.google.com/search?q=PID+control+theory
http://en.wikipedia.org/wiki/PID_controller
PID without a PhD

Programming Techniques

EV3:

To use capabilities of the sensor, please download EV3 blocks available at following URL:
http://www.mindsensors.com/index.php?controller=attachment&id_attachment=192

Installation instructions for EV3 block are available at:
http://www.mindsensors.com/content/13-how-to-install-blocks-in-ev3

Download EV3 sample program from following URL and modify it to suit your needs.
http://www.mindsensors.com/index.php?controller=attachment&id_attachment=193

NXT-G:

Download the NXT-G blocks available in the NXT-G Blocks Repository at mindsensor's website, at following location:
http://www.mindsensors.com/index.php?controller=attachment&id_attachment=195

Installation instructions for NXT block are available at:
http://www.mindsensors.com/content/21-nxt-g-blocks-how-to-install-blocks

Also download sample programs from following location, and modify to suit your needs.
http://www.mindsensors.com/index.php?controller=attachment&id_attachment=196

RobotC:

The driver implementation is available in Xander's driver suite at following url:

https://github.com/botbench/robotcdriversuite

the header file for the driver is: mindsensors-lightsensorarray.h

example program: mindsensors-ligthsensorarray-test1.c

NXC:

Download the sample programs and library file available at following location, and include the library file it in your program by #include directive as:

#include "LSA-lib.nxc"

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=191

Alternately, you can modify the sample programs to suite your needs.

I2C Registers:

The LightSensorArray appears as a set of registers as follows:

Register	Read	Write
0x00-0x07	Firmware version – *Vxxxx*	-
0x08-0x0f	Vendor Id – *mndsnsrs*	-
0x10-0x17	Device ID – LSArray	-
0x41	-	Command
Register	Read	Write
0x42 – 0x49	Calibrated Sensor reading - Reading measured for each of the sensor in the array. Higher value indicates brighter object. (one byte for each sensor – 8 bytes)	-
0x4A – 0x51	White Reading Limit (threshold above which color is identified as white)	-
0x52 – 0x59	Black Reading Limit (threshold below which color is identified as black)	-
0x5A-0x61	White Calibration data - Calibration value for White color for each sensor. (one byte for each sensor – 8 bytes)	-
0x62-0x69	Black Calibration data - Calibration value for Black color for each sensor (one byte for each sensor – 8 bytes).	-

0x6A-0x79	Uncalibrated sensor voltage (Integer values, two bytes for each sensor)	-

Supported I2C Commands:

CMD	Description
W	Calibrate White
B	Calibrate Black
D	Put Sensor to sleep
P	Wake up the sensor
A	Configure Sensor for US region (and regions with 60 Hz electrical frequency).
E	Configure sensor for European region (and regions with 50 Hz electrical frequency)
U	Configure sensor for universal frequency (in this mode the sensor adjusts for any frequency, this is also the default mode).

Reading Frequency

The array of sensors is reading at 7 milliseconds interval.
Note however, your reading speed from sensor is throttled by the I2C speed of your NXT firmware (in firmware 1.29 it is around 17 milliseconds per reading operation).

Physical Specs

Weight: 0.53 oz (14.8 grams)
Foot-print: 24.5mm x 72.6 mm
Height: 20.75 mm

Current Consumption

To conserve power, the sensor goes to sleep after 1 minute of inactivity. The sensor will wake up on its own when any activity begins. You can also wake it up (or put to sleep) via command. Average measured current profile is as follows:

Current Consumption	Duration
5.7 mA (max)	While awake Depending on your mat colors ratio, this will vary between 3.5 mA to 5.7 mA
3.0 mA	While sleeping

I2C Bus address

Factory Default Address: 0x14 (Decimal: 20)

Changing the I2C Bus Address:

The I2C bus address of LightSensorArray can be changed. To set an address different from default address, send sequence of following commands on the command register:

0xA0, 0xAA, 0xA5, <new I2C address>

Note: Send these commands with no break/read operation in between. This new address is effective immediately. Please note down your address carefully for future reference.

Instructions for changing the I2C address can be found at:

NXT and EV3
http://www.mindsensors.com/blog/how-to/change-i2c-device-address.

PiStorms
http://www.mindsensors.com/blog/how-to/change-i2c-device-address-with-pistorms

Doing your own PID Control

If you wish to write your own PID control, following urls and references would be useful.
An excellent reference describing PID control:
http://en.wikipedia.org/wiki/PID_controller

A Google search on PID Control theory yields several informative artifacts -
http://www.google.com/search?q=PID+control+theory

An article from Embedded Systems: PID without a PhD
(http://www.embedded.com/2000/0010/0010feat3.htm)