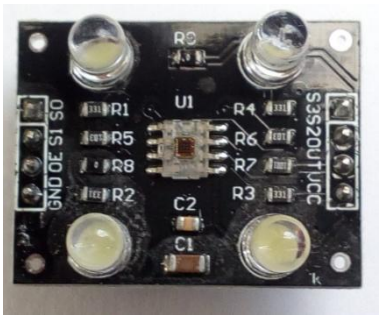
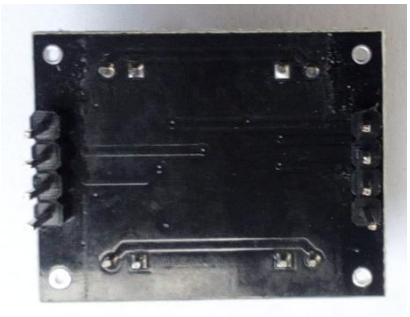
55 - TCS230 sensore di colore – color sensor (some notes at section end)





Il sensore di colore e' un dispositivo composto da un rilevatore TCS230 e da quattro led bianchi. Il rilevatore TCS230 e' a sua volta composto da una matrice di 64 fotorilevatori: 16 sensibili al colore verde, 16 sensibili al colore blu, 16 al rosso e 16 neutri, distribuiti in modo uniforme per eliminare o ridurre eventuali errori di valutazione dovuti alla loro. Il sistema e' anche dotato di un oscillatore in grado di produrre, in uscita, un segnale ad onda quadra la cui frequenza (da 2 hertz a 500 khertz) e' proporzionale all'intensita' del colore rilevato dai fotorilevatori.

Il dispositivo e' dotato di ingressi ed uscite digitali che consentono un collegamento diretto ad Arduino. I piedini S2 e S3 sono usati per selezionare il gruppo di fotorilevatori (rosso, verde, blu e neutro) da attivare di volta in volta mentre i piedini S0 ed S1 consentono di selezionare il range su cui l'oscillatore calibra le frequenze in uscita (per consentirne l'utilizzo su diversi tipi di microcontrollori).

Il sistema produce, in uscita, un segnale ad onda quadra la cui frequenza e' compresa nella "scala" selezionata dai piedini S0 ed S1 ed è proporzionale all'intensità del colore rilevato dai fotorilevatori selezionati con i piedini S2 ed S3. Queste le possibili scelte effettuabili agendo sui piedini S0, S1, S2, ed S3:

Range frequenze in uscita	S0	S1
Oscillatore spento	LOW	LOW
2%	LOW	HIGH
20%	HIGH	LOW
100%	HIGH	HIGH

Filtri selezionabili	S2	S3
Rosso	LOW	LOW
blu	LOW	HIGH
Nessun filtro	HIGH	LOW
Verde	HIGH	HIGH

Con Arduino e' possibile utilizzare sia frequenze in uscita pari al 100% che quelle al 20% che quelle al 2%. Questo progetto, che si limita ad esporre i valori e le conclusioni sul monitor seriale, utilizza il range del 20%.

In pratica, un oggetto posizionato davanti al sensore, ad una distanza non superiore a 10 mm, viene illuminato dai 4 led bianchi e la luce riflessa colpisce i 64 fotorilevatori producendo in uscita, per il colore selezionato tramite i pin S2 ed S3, un'onda quadra la cui frequenza e' direttamente proporzionale all'intensità della luce riflessa.

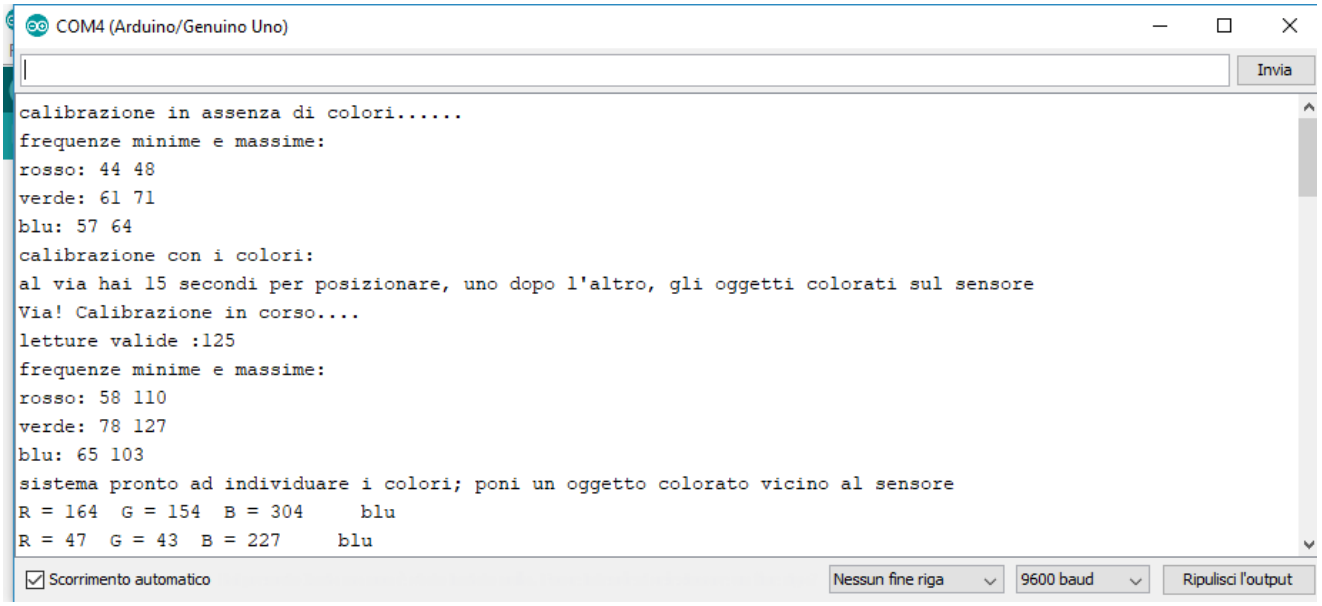
In questo progetto ci si limita a convertire in valori RGB (da 0 a 255) i segnali prodotti dal sensore e con questi si tenta di individuare la famiglia di colore (rosso, verde o blu) cui appartiene l'oggetto posto davanti al sensore.

Non viene tentata l'individuazione dei colori secondari (come il giallo, l'arancione o il viola) poiche' la variabilita' (o meglio l'attendibilita') dei valori primari forniti dal sensore non consente alcuna ulteriore analisi.

Per chi volesse approfondire il tema, questo sito consente di osservare quanto i colori secondari siano complessi da definire in termini di valori RGB: https://www.geogebra.org/m/YeVXW9qF

Per ottenere valori stabili e per quanto possibile vicini ai reali valori RGB, il programma prevede una fase di calibrazione, che dura una ventina di secondi, durante la quale l'operatore deve, su richiesta del sistema, avvicinare al lettore per qualche secondo (due o tre secondi) gli oggetti di cui intende far individuare il colore. Terminata la fase di calibrazione il sistema espone, sul monitor la frase "*sistema pronto ad individuare i colori; poni un oggetto colorato vicino al sensore*", e da quel momento e' pronto a individuare i colori che verranno via via avvicinati.

Qui il filmato del progetto: TCS230 sensore di colore



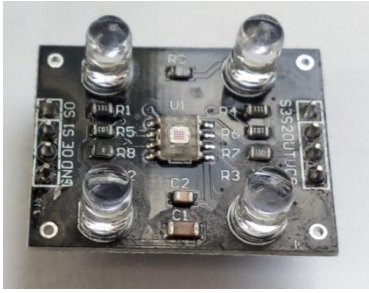
Attenzione: se durante la fase di calibrazione vengono segnalati dei valori fuori scala, sara' necessario aumentare la luce ambientale e ripetere la calibrazione, reinizializzando Arduino.

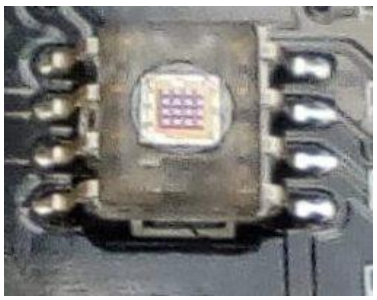
Nota: Questo esercizio e questa nota sono parte di una serie che vede protagonisti Arduino ed alcuni dei componenti ad esso collegabili. Per la maggior parte degli esercizi e' anche disponibile un filmato su youtube:

- Esercizi facenti parte della raccolta
- Filmati presenti su youtube
- Informazioni su arduino e sui componenti collegabili (PDF scaricato nell'area di download)
- Breve manuale di programmazione (PDF scaricato nell'area di download)

Per eventuali chiarimenti o suggerimenti sul contenuto di questa scheda scrivere a giocarduino@libero.it

Here some notes about this project, translated by google translator





The color sensor is a device composed by a TCS230 detector and four white leds. Detector is in turn composed by a 64 photodiodes matrix: 16 sensitive to green color, 16 sensitive to blue, 16 to red and 16 neutrals distributed evenly, to eliminate or reduce any errors of evaluation due to their position. The system is also equipped by an oscillator, able to produce a square wave signal whose frequency (from 2 hertz to 500 khertz) is proportional to the color detected by the photodiodes.

Device is equipped with some digital inputs and outputs pins, that allow it a direct connection to Arduino. Pins S2 and S3 are used to select the group of photodiodes (red, green, blue, neutral) to be activated time by time; pins S0 and S1 allow to select the scale factor on which the oscillator calibrates the output frequencies (to allow its use on different types of microcontrollers).

System produces a square wave whose frequency is included in "scale" selected by pins S0 and S1 and is proportional to the intensity of color detected color by photodiodes selected by pins S2 and S3. These are all the possible choices that can be made by acting on pins S0, S1, S2, and S3:

Output frequency range	S0	S1
Oscillator off	LOW	LOW
2%	LOW	HIGH
20%	HIGH	LOW
100%	HIGH	HIGH

Selectable filters	S2	S3
red	LOW	LOW
blue	LOW	HIGH
No filter (neutral)	HIGH	LOW
green	HIGH	HIGH

Arduino can manages all output frequencies, and in this project, which is limited to displaying values and conclusions on serial monitor, uses a reduced range to 20%.

Pratically, an object placed in front of sensor at about 10 mm distance, is illuminated by the 4 white leds and the reflected light affects the 64 photodiodes producing in output, for color selected by S2 and S3 pins, a square wave whose frequency is directly proportional to the intensity of reflected light.

In this project we convert the signals produced by sensor into RGB values (from 0 to 255) and with these we try to identify the color family (red, green or blue) to which the object placed in front of the sensor belongs.

The identification of secondary colors (such as yellow, orange or violet) is not attempted because of the variability (or rather the reliability) of the primary values provided by sensor that seems does not allow any further analysis.

For who wishing to deepen the theme, in this site, acting on colored arrows, can verify how the secondary colors are complex to define in terms of RGB values:
https://www.geogebra.org/m/YeVXW9qF

To obtain stable values and as close as possible to the real RGB values, program includes a "setting" phase, which lasts about twenty seconds, during which, at system request, operator must put near sensor, for a few seconds (two or three seconds), the objects of which he intends to identify color. Once the setting phase is over, the system displays " *sistema pronto ad individuare i colori; poni un oggetto colorato vicino al sensore*" (*system ready, put a colored object near sensor*), and from that moment it is ready to identify colors that will be approached.

Here the project movie (subtitled): TCS230 – color sensor

Warning: if, during the setting phase, a "*fuori scala*" (*out o range*) phrase is displayed, it will be necessary to increase ambient light and repeat the setting (restart Arduino).

Note: This project and this note is part of a series that sees, as main characters, Arduino and some of connectable components. For most projects there is also a video on youtube.

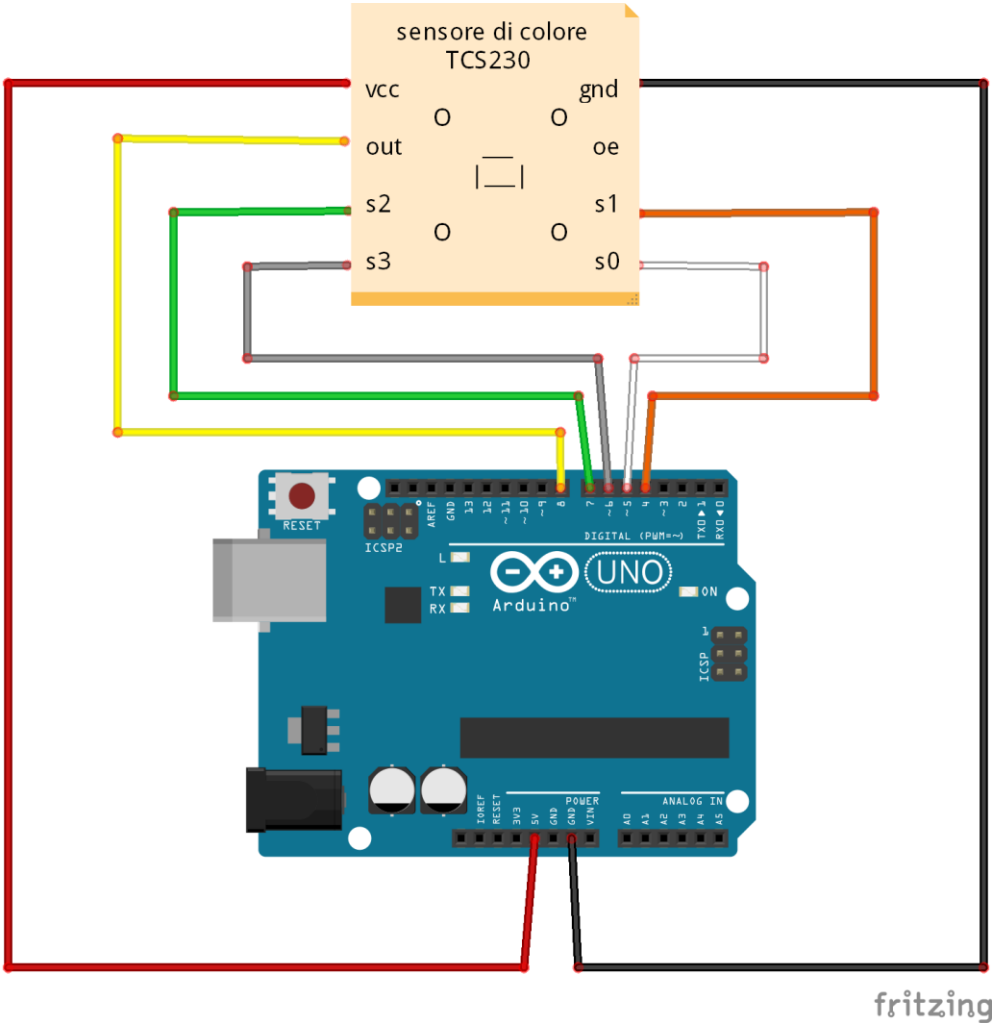
- Projects collection
- Movies on youtube
- About Arduino and components (italian; pdf will be downloaded in your download area
- Quick programming guide (almost english; pdf will be downloaded in your download area)

For any questions or suggestions about this note (and on its english translation), please write to giocarduino@libero.it (simple words and short sentences, please).

Materiali

- Un sensore di colore modello TCS230

Schema



Programma

```
/*
  Attenzione: facendo il copia/incolla dal PDF all'IDE si perde la formattazione del testo. Per
  rendere piu' facilmente leggibile il programma e' opportuno formattarlo subito dopo il
  trasferimento nell'IDE, premendo CTRL+T.

  Questo programma identifica il colore di un oggetto (rosso o verde o blu) posto davanti ad un
  sensore di tipo TCS230.
  La luce riflessa dall'oggetto viene "letta" dal sensore utilizzando, in rapida
  sequenza, diodi sensibili al colore rosso, al colore verde ed al colore blu. Il sensore trasforma
  ogni lettura in una serie di segnali digitali, la cui frequenza e' inversamente proporzionale al
  colore dell'oggetto in esame. Arduino converte queste frequenze (rilevate tramite l'istruzione
  "pulseIn") in un valore compreso tra 0 a 255 e poi le analizza, tentando di interpretarle.

  Per ottenere valori stabili, i parametri di mappatura (ciclo di while e istruzione map delle
  frequenze segnalate da ogni colore) devono essere Definiti di volta in volta, in funzione della
  luce ambientale e della intensita' del colore degli oggetti campione.
```

```
  Nella fase iniziale il programma esegue quindi una serie di attivita' di calibrazione, necessarie
  alla definizione dei parametri sia in assenza di oggetti che in presenza di oggetti colorati

  Al termine della fase di calibrazione il sistema disporra' dei parametri necessari al
  riconoscimento di ogni colore.

  Una volta stabiliti i parametri (e cioe' i valori minimi e massimi di ogni foto rilevatore), in
  presenza di colori il programma, sempre per rendere stabile e sicura l'individuazione del colore,
  esegue 5 letture consecutive per ogni oggetto posto vicino al sensore ed alla fine espone
  e considera la media dei valori rilevati.
  ---------------------------------------------------------------------------------------------------
   Warning: cut&paste from PDF to IDE loses formatting. To restore it press CTRL + T.

  This program identifies a color of an object (red or green or blue) placed near a TCS230 sensor.

  More in detail, the light reflected by object is "read" by sensor using, in rapid succession,
  diodes sensitive to red, green and blue.
  Sensor transforms each reading into a series of digital signals, whose frequency is inversely
  proportional to the color of the object . Arduino converts this frequency (detected by the
  "pulseIn" instruction) in a value between 0 and 255 and then analyzes frequencies proposed by the
  "reading" performed by the three different types of diodes, and tries to interpret them.

  To obtain stable values and close to the real RGB values, the mapping parameters ("while" cycle
  and "map) statement)  must be defined each time, depending on ambient light and color intensity of
  the sample objects.
  In the initial phase, program performs a series of activities for defining the parameters both in
  the absence of objects and in presence of colored objects.

  At the end of the initialization phase, system will have defined the parameters necessary for the
  recognition of each color.

  Once the parameters have been established (and that is the minimum and maximum values reeased by
  each photodetector), program performs 5 consecutive readings for each object placed near the
  sensor and then exposes and considers the average of the values measured.

*/

#define S0 5
#define S1 4
#define S2 7
#define S3 6
#define sensore 8
int frequenza    = 0;
int freqrosso    = 0;
int freqverde    = 0;
int freqblu      = 0;
int rosso        = 0;
int blu          = 0;
int verde        = 0;
int indice       = 0;
byte tabvalori [751];    // tabella memorizzazione delle ripetizioni delle frequenze di ogni colore;
// l'indice della tabella e' la frequenza, e le frequenze analizzate vanno da 0 a 250 per tre colori
// matrix to storing frequencies repetitions for each color; the matrix index is the frequency, and
// the analyzed frequencies go from 0 to 250, for three colors
//
float tempoiniz   = 0;   // momento di inizio calibrazione
float tempocorr  = 0;    // momento attuale
byte minaccett   = 0;    // Occorrenze minime accettabili (il 2 % del numero totale di occorrenze)
byte minrosso    = 250;
byte maxrosso    = 0;
byte minverde    = 250;
byte maxverde    = 0;
byte minblu      = 250;
byte maxblu      = 0;
int contatore    = 0;
byte minrosson   = 250;
byte maxrosson   = 0;
byte minverden   = 250;
byte maxverden   = 0;
byte minblun     = 250;
byte maxblun     = 0;

//
//**** routine di lettura delle frequenze in uscita dal sensore ****
//
void lettura ()
{
```

```
  //********* rileva la frequenza emanata dai fotorilevatori sensibili al rosso *******************
  // ***************detects frequency emitted by red sensitive photodetectors *********************
  digitalWrite(S2, LOW);
  digitalWrite(S3, LOW);
  frequenza = pulseIn(sensore, LOW);
  if (frequenza > 250)
  {
    Serial.print ("fuori scala rosso: ");
    Serial.println (frequenza);
  }
  else
    freqrosso = frequenza;
  delay (30);
  //********* rileva la frequenza emanata dai fotorilevatori sensibili al verde *******************
  //***************detects frequency emitted by green sensitive photodetectors *********************
  digitalWrite(S2, HIGH);
  digitalWrite(S3, HIGH);
  frequenza = pulseIn(sensore, LOW);
  if (frequenza > 250)
  {
    Serial.print ("fuori scala verde: ");
    Serial.println (frequenza);
  }
  else
    freqverde = frequenza;
  delay (30);
  //********* rileva la frequenza emanata dai fotorilevatori sensibili al blu *******************
  //***************detects frequency emitted by blue sensitive photodetectors *******************
  digitalWrite(S2, LOW);
  digitalWrite(S3, HIGH);
  frequenza = pulseIn(sensore, LOW);
  if (frequenza > 250)
  {
    Serial.print ("fuori scala blu: ");
    Serial.println (frequenza);
  }
  else
    freqblu = frequenza;
  delay (30);
}

void setup()
{
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(sensore, INPUT);
  //
  // sets sensor for output a range equal to 20% of the frequencies
  digitalWrite(S0, HIGH);  // setta il sensore (pin s0 ed s1) in modo che fornisca un segnale
  digitalWrite(S1, LOW);   // la cui frequenza sia pari al 20& della frequenza massima fornibile
  Serial.begin(9600);
  //
  // ********** fase di calibrazione sulle frequenze minime e massime **********
  // *************** calibration phase on minimum and maximum frequencies ******
  //
  // ***********  calibrazione in assenza di colori ************************************
  // *******************calibration without colors ************************************
  //
  Serial.println ("calibrazione in assenza di colori......"); // calibration without colors...
  for (indice = 1; indice < 50; indice ++)
  {
    lettura ();             // lancia la routine di lettura delle frequenze rilasciate dal sensore
    tabvalori [freqrosso * 3] = tabvalori[freqrosso * 3] + 1; // conteggia la frequenza del rosso
    tabvalori [(freqverde * 3) + 1] = tabvalori[(freqverde * 3) + 1] + 1; // conteggia freq. verde
    tabvalori [(freqblu * 3) + 2] = tabvalori[(freqblu * 3) + 2] + 1; // conteggia frequenza blu
  }
  //
  // *************** uscito dalla fase di rilevazione, analizza i dati memorizzati ***************
  // ************************  analyzes stored data  ********************************************
  //
  minaccett = 1;    // definisce la minima occorrenza accettabile (il 2% del numero di cicli)
  for (indice = 1; indice < 250; indice ++)  // analizza la tabella delle frequenze per individuare
  // la frequenza minima e massima rilevata dai fotorilevatori di ogni colore
  {
    if (tabvalori [indice * 3] > minaccett) // se il numero di occorrenze rosso e' > del minimo
    //  if red occurrences is greater than minimum acceptable
```

```
    {
      if (indice < minrosso)
        minrosso = indice;   // memorizza la frequenza (l'indice e' la frequenza, mentre in tabella
      // ci sono le occorrenze) - memorizes frequency (index is the frequency, while in matrix there
      // are occurrences)
      if (indice > maxrosso)
        maxrosso = indice;
    }
    if (tabvalori [(indice * 3) + 1] > minaccett) // se le occorrenze verde sono superiore al minimo
    {
      if (indice < minverde)
        minverde = indice;    // memorizza la frequenza (l'indice e' la frequenza, mentre in tabella
        //  ci sono le occorrenze)
      if (indice > maxverde)
        maxverde = indice;
    }
    if (tabvalori [(indice * 3) + 2] > minaccett) // se le occorrenze blu sono superiore al minimo
    {
      if (indice < minblu)
        minblu = indice;     // memorizza la frequenza (l'indice e' la frequenza, mentre in tabella
        // ci sono le occorrenze)
      if (indice > maxblu)
        maxblu = indice;
    }
  }
  Serial.println ("frequenze minime e massime:");  // minimum and maximum frequencies
  Serial.print ("rosso: ");                         // red
  Serial.print (minrosso);
  Serial.print (" ");
  Serial.println (maxrosso);
  Serial.print ("verde: ");                        // green
  Serial.print (minverde);
  Serial.print (" ");
  Serial.println (maxverde);
  Serial.print ("blu: ");                         // blue
  Serial.print (minblu);
  Serial.print (" ");
  Serial.println (maxblu);
  for (indice = 0; indice < 750; indice ++)
    tabvalori [indice] = 0;  // azzera la tabella tabvalori
  // memorizza le frequenze minime e massime in assenza di colore (aumentati o diminuiti di 2 unita)
  // memorizes  minimum and maximum frequencies in absence of color (increased or decreased by 2)
  minrosson = minrosso - 2;
  maxrosson = maxrosso + 2;
  minverden = minverde - 2;
  maxverden = maxverde + 2;
  minblun   = minblu - 2;
  maxblun   = maxblu + 2;
  // ripristina i valori iniziali per una nuova calibrazione - restores the initial values for a new
  // calibration
  minrosso = 250;
  maxrosso = 0;
  minverde = 250;
  maxverde = 0;
  minblu = 250;
  maxblu = 0;
  Serial.println ("calibrazione con i colori: ");  // colors calibration:
  Serial.println ("al via hai 15 secondi per posizionare, uno dopo l'altro, gli oggetti colorati sul sensore");
  // at "via" you have 15 seconds to place, one after another, the colored objects near sensor
  delay (3000); // aspetta tre secondi, per consentire la lettura dei dati esposti
  Serial.println ("Via! Calibrazione in corso...."); // "via! calibration in progress...
  tempoiniz = millis ();
  tempocorr = millis ();
  while ((tempocorr - tempoiniz) < 15000)  // se non sono passati 15 secondi
  {
    tempocorr = millis ();     //  memorizza il momento corrente
    lettura ();                // lancia la routine di lettura delle frequenze rilasciate dal sensore
//
// Attenzione, quella che segue e' un'unica istruzione: se appare spezzata su due righe deve
//essere eliminato il segnale di fine riga - Warning, the following is a single instruction: if it
//appears broken on two lines the end-of-line signal must be eliminated
    if (((( freqrosso < minrosson) || (freqrosso > maxrosson)) && ((freqverde < minverden) ||
(freqverde > maxverden)) && ((freqblu > minblun) || (freqblu < maxblun))))
    { // se la lettura e' valida (se il sensore ha letto un colore) - if reading is ok (if sensor
      // has read a color)
      tabvalori [freqrosso * 3] = tabvalori[freqrosso * 3] + 1; // conteggia la frequenza del rosso
      // add 1 to red frequency occurencies
      tabvalori [(freqverde * 3) + 1] = tabvalori[(freqverde * 3) + 1] + 1; // conteggia la
```

```
    // frequenza del verde - add 1 to green frequency occurencies
    tabvalori [(freqblu * 3) + 2] = tabvalori[(freqblu * 3) + 2] + 1;     // conteggia la
    // frequenza del blu - add 1 to blue frequency occurencies
    contatore ++;                           // conta le letture valide - counts valid readings
  }
}
//
// *************** uscito dalla fase di rilevazione, analizza i dati memorizzati ***************
// ************************  analyzes stored data  ********************************************
//
Serial.print ("letture valide :");  // valid readings:
Serial.println (contatore);
minaccett = contatore / 50; // definisce la minima occorrenza accettabile (2% del numero di cicli)
for (indice = 1; indice < 250; indice ++)  // analizza la tabella delle frequenze per individuare
// la frequenza minima e massima rilevata dai fotorilevatori di ogni colore
{
  if (tabvalori [indice * 3] > minaccett) // se le occorrenze del rosso sono superiori al minimo
  //  if red occurrencies is greater than the minimum acceptable
  {
    if (indice < minrosso)
      minrosso = indice;     // memorizza la frequenza (l'indice e' la frequenza, mentre in
     // tabella sono presenti le occorrenze -  memorizes frequency (index is the frequency, while
     // in matrix there are occurrences)
    if (indice > maxrosso)
      maxrosso = indice;
  }
  if (tabvalori [(indice * 3) + 1] > minaccett) // se le occorrenze verde sono superiori al minimo
  {
    if (indice < minverde)
      minverde = indice;     // memorizza la frequenza (l'indice e' la frequenza, mentre in
     // tabella sono presenti le occorrenze
    if (indice > maxverde)
      maxverde = indice;
  }
  if (tabvalori [(indice * 3) + 2] > minaccett) // se le occorrenze blu sono superiori al minimo
  {
    if (indice < minblu)
      minblu = indice;     // memorizza la frequenza (l'indice e' la frequenza, mentre in tabella
     // sono presenti le occorrenze
    if (indice > maxblu)
      maxblu = indice;
  }
}
//
// allarga il range delle frequenze minime e massime utilizzate per individuare i colori
// widens range for minimum and maximum frequencies used to identify each color
if (minrosso > 3)
  minrosso = minrosso - 3;
else
  minrosso = 0;
if (maxrosso < 247)
  maxrosso = maxrosso + 3;
else
  maxrosso = 250;
if (minverde > 3)
  minverde = minverde - 3;
else
  minverde = 0;
if (maxverde < 247)
  maxverde = maxverde + 3;
else
  maxverde = 250;
if (minblu > 3)
  minblu = minblu - 3;
else
  minblu = 0;
if (maxrosso < 247)
  maxblu = maxblu + 3;
else
  maxblu = 250;
Serial.println ("frequenze minime e massime:");  // minimum and maximum frequencies for each color
Serial.print ("rosso: ");
Serial.print (minrosso);
Serial.print (" ");
Serial.println (maxrosso);
Serial.print ("verde: ");
Serial.print (minverde);
Serial.print (" ");
```

```
  Serial.println (maxverde);
  Serial.print ("blu: ");
  Serial.print (minblu);
  Serial.print (" ");
  Serial.println (maxblu);
  Serial.println ("sistema pronto ad individuare i colori; poni un oggetto colorato vicino al sensore");
  // system ready; put a colored object near sensor
}
//
//
void loop()
{
  rosso = 0;
  verde = 0;
  blu = 0;
  tempoiniz = millis ();  // memorizza il momento di inizio del loop delle 5 letture
  for (indice = 1; indice < 6; indice = indice)  // esegue 5 letture  e utilizza la media
  // per individuare il colore - performs 5 readings and uses average to identify color
  {
    tempocorr = millis (); // memorizza il momento della rilevazione
    if ((tempocorr - tempoiniz ) > 2000)  // se sono passati piu' di due secondi da inizio
    // rilevazione significa che il sensore non vede alcun colore e quindi azzera i dati e
    // ricomincia il loop di lettura - if more than two seconds have elapsed since the
    // detection beginning, it means that sensor does not see any color and therefore resets
    // data and starts the reading loop again
    {
      indice = 1;
      tempoiniz = tempocorr;
      rosso = 0;
      verde = 0;
      blu = 0;
    }
    else     // se sta leggendo un colore - if is reading a color
    {
      lettura (); // lancia la routine di lettura dei colori
   //
   // Attenzione, quella che segue e' un'unica istruzione: se appare spezzata su due righe deve
   //essere eliminato il segnale di fine riga - Warning, the following is a single instruction: if
   // it appears broken on two lines the end-of-line signal must be eliminated
      if (((( freqrosso < minrosson) || (freqrosso > maxrosson)) && ((freqverde < minverden) ||
(freqverde > maxverden)) && ((freqblu > minblun) || (freqblu < maxblun))))
   // se la lettura e' valida (se il sensore ha individuato un oggetto) - if reading is valid
      {
        indice++;
      // rimappa la frequenza per conventirla in valori RGB - convert frequencies in an RGB value
       rosso = rosso + map(freqrosso, minrosso, maxrosso, 255, 0);
       verde = verde + map(freqverde, minverde, maxverde, 255, 0);
       blu = blu + map(freqblu, minblu, maxblu, 255, 0);
      }
    }
  }

  // ********** individua il colore *********
  //  ********* compute a color **************
  Serial.print("R = ");
  Serial.print (rosso / 5);
  Serial.print("  ");
  Serial.print("G = ");
  Serial.print (verde / 5);
  Serial.print("  ");
  Serial.print("B = ");
  Serial.print (blu / 5);
  Serial.print("  ");
  if ((blu > verde) && (blu > rosso))
    Serial.println ("   blu");  // blue
  if ((verde > blu) && (verde > rosso))
    Serial.println ("   verde");  // green
  if ((rosso > verde) && (rosso > blu))
    Serial.println ("   rosso");   // red
}
```