# Dosage Colorimétrique Arduino

Esma MOCTAR SALEM; Ghida AL KHALIL; Imene HAMAZ; Iyad CHAHINE; Victor ELEONORE

Projet E

Le **dosage** est une technique permettant de déterminer la quantité de matière ou la concentration d'une entité chimique dans une solution connue. Les réactions de dosage sont des réactions rapide, totale et ou le réactif de la solution titrante (solution de concentration connue) doit réagir avec une unique espèce d'un solution titrée (solution de concentration inconnue). Le dosage colorimétrique est une technique permettant de repérer l'équivalence par un changement de couleur du milieu réactionnel. Le changement de couleur de la solution également que les réactifs ont été introduits dans les proportions stœchiométriques.

L'objectif du projet est de réguler et d'automatiser un dosage colorimétrique acido-basique par Arduino. Il s'agira de titré de l'acide chlorydrique (HCl), par une solution de soude (NaOH) solution titrante de concentration connue. L'indicateur colorée utilisé est la phénolphtaléine. Pour réaliser le projet, le matériel suivant est utilisé:

- Burette
- Béchers
- Servomoteur
- Une photorésistance (pour mesurer l'intensité lumineuse)
- Capteur de couleur TCS34725
- une LED pour mettre en évidence le changement de couleur
- 2 Résistances
- Microcontrôleur Arduino
- BreadBoard
- phénolphtaléine (indicateur coloré)

## I. Matériel

1. Arduino

La première partie du projet consiste à réaliser les branchements le moteur pas à pas, la LED, la photorésistance et les résistances sur le microcontrôleur Arduino. Après avoir réalisé un schéma de câblage sur Tinkercad, nous avons branché nos différents équipements sur Arduino.

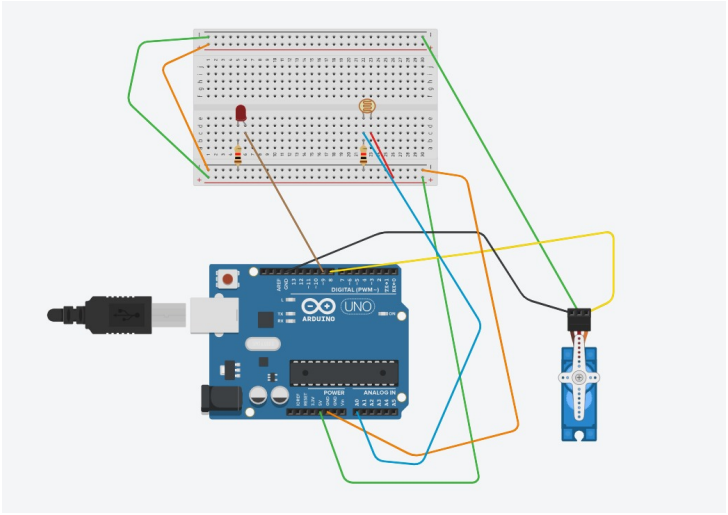

Figure 1: Schéma de câblage sur Tinkercad

Après avoir branché nos appareils sur Arduino, différents langage informatique ont été mis sur Arduino afin de montrer que les différents équipements pouvait fonctionner indépendamment des autres. Il a donc été démontré que les appareils fonctionnait indépendamment des autres, cependant certains dysfonctionnement ont pu être mis en avant.:

- Le moteur pas à pas qui se bloque après avoir atteint une valeur seuil.

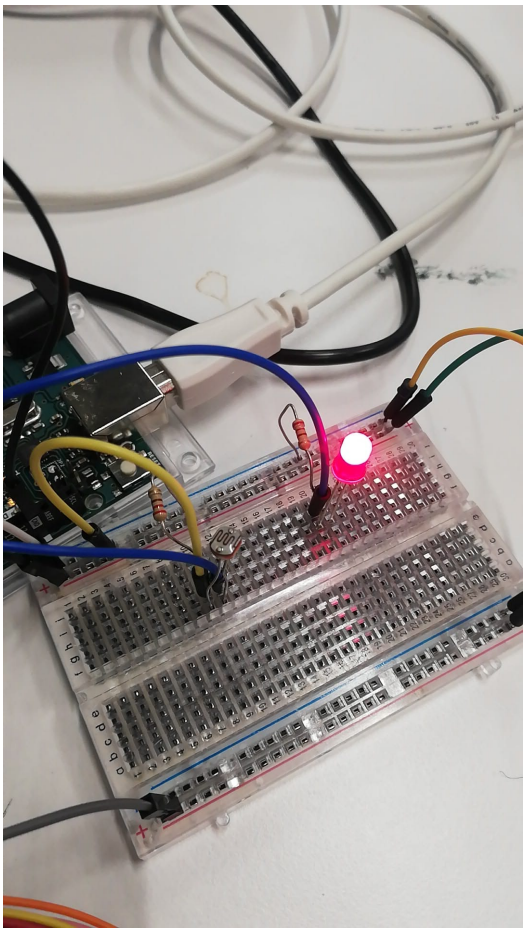

Figure 2: Test de fonctionnement de la LED

Nous avons trouvé un servo moteur mieux adapté à la plaque Arduino ainsi qu'aux pièces que nous possédons qui relient le moteur à la clé de robinet de la burette. Ainsi, voici notre montage:

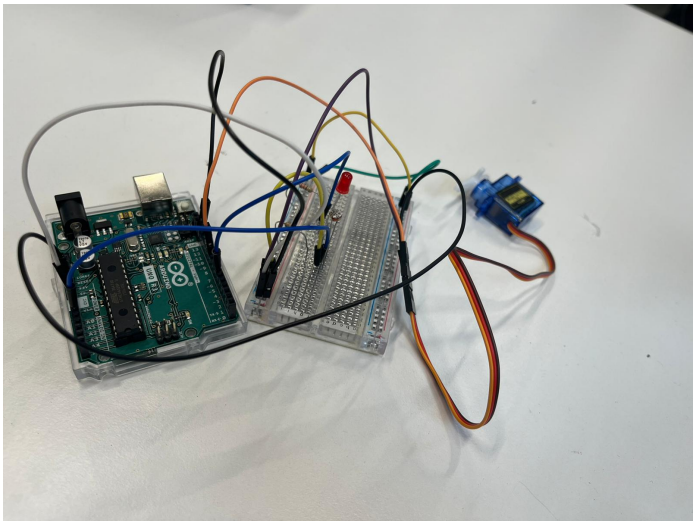

Figure 3: Montage de l'Arduino

Après avoir rentré un langage informatique sur Arduino pour nous assurer que le servomoteur fonctionnait. Le code que nous avons rentré consistait à faire tourner les pales du servomoteur en fonction de la valeur de l'angle (variables que nous rentrons dans le langage informatique). Après plusieurs essais, nous avons pu constater que le servomoteur fonctionnait et qu'il était apte à faire tourner le robinet de la burette.

Enfin, nous avons vérifier l'état de fonctionnement de la photorésistance. Pour cela nous avons utiliser le code suivant pour la faire fonctionner :

```
int photo = A0; // Déclaration de la photorésistance sur Arduino
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(photo, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  int lecture = analogRead(photo); // Lecture de la valeur enregistré par la photorésistance en fonction de l'intensité lumineuse
  Serial.println(lecture); // Enregistrement la valeur sur Arduino (Les valeurs sont enregistrés en ligne
}
```

Ce code nous à permis de calibrer la photorésistance en fonction de l'intensité lumineuse. Ainsi, nous avons pu établir une corrélation entre les valeurs enregistrés par Arduino, et l'intensité lumineuse.

| Conditions dans laquelle se trouve la photorésistance | Valeur enregistré par Arduino | Conclusion |
|---|---|---|
| -Boite en carton fermé | Valeur aux alentours de 10 | La photorésistance ne capte aucune lumière |
| -Boite en carton en fermé troué pour faire passer un faisceau lumineux | Valeur aux alentours de 170 | |
| -Photorésistance éclairé par une lampe torche de téléphone à une distance d'environ 3 à 5 cm | Valeurs aux alentours de 975 | |
| -Lampe torche du téléphone collé à la photorésistance | Valeurs supérieur à 1013 | On est presque au maximum de l'intensité lumineuse pouvant être capté par la photorésistance |
| -Photorésistance éclairée par la lumière ambiante du FabLab à 17h13 | Valeurs aux alentours de 866 | |

2.Impression 3D et Construction d'une boîte opaque

Pour pouvoir tourner le robinet de la burette, un servomoteur n'est pas suffisant. En effet, nous avons dû adapter une clé de robinet sur la burette. Cette clé sera scotché aux pales du servomoteur afin de pouvoir faire tourner le robinet de la burette, et de verser la solution titrante dans la solution titrée. La clé de robinet à été réalisé à l'aide de l'impression 3D en utilisant du PFA.

On a commencé par la réalisation du modèle 3D à l'aide du site "Onshape", un logiciel/site de représenté sur Onshape :

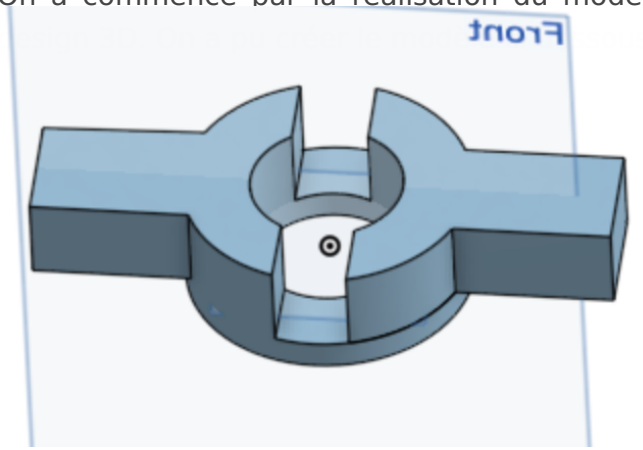


Figure 4: Modèle 3D de la Clé de robinet

Les dimensions ont été adaptés à la clé de robinet et au servomoteur qui sera lié à la clé. On note une longueur de 55 cm, et un diametre interne de 10,75 cm. On obtient par impression 3D le modèle suivant :

Figure 5: Clé de robinet

Dans le cadre du dosage il est nécessaire de mettre le bécher dans une boîte opaque pour que les valeurs affichés par la photorésistance ait du "sens". Autrement dit, il faut que la photorésistance capte uniquement la lumière issu de la source lumineuse (dans notre cas il s'agit d'une lampe de poche), et pas la lumière produit par les lumière de la salle ou de la hotte.

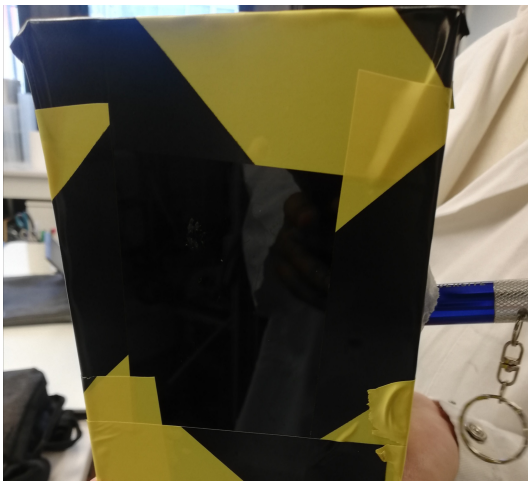

Figure 6: Boîte opaque

3.Code Arduino

En ce qui concerne le code Arduino, nous avons trouvé des codes séparés pour la LED (Fig ), la photorésistance (Fig ) et le servomoteur (Fig ), que nous avons testé afin de vérifier le bon fonctionnement du matériel. Le montage est installé, nous avons ensuite écris un code adapté à faire tourner notre système en prenant compte des trois unités simultanément.

```
#include <Servo.h>  // on inclut la bibliothèque servo

Servo servoMoteur;   // on crée un objet servo appelé servoMoteur
```

```
//set pin numbers
//const won't change
const int ledPin = 5;   //Déclaration de la LED sur ARDUINO
const int ldrPin = A0;  //Déclaration de la photorésistance


void setup() {

  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);  //iLED en sortie
  pinMode(ldrPin, INPUT);   //Photorésitance en entrée
  servoMoteur.attach(6); // Servo moteur en position 9
}

void loop() {

  int ldrStatus = analogRead(ldrPin);   //Lecture de la valeur de la photorésistance
  //Regarder si la valeur de la photorésistance est inferieur ou égale à <= x
  //Si c'est le cas, la LED est éteinte le titrage commence

   if (ldrStatus <=x) {

    digitalWrite(ledPin, LOW);              //LED est eteinte
    Serial.println();                     //Les valeurs de la photorésitance sont enrgistrés en ligne
    servoMoteur.write(0);                   //Le servomoteur passe de l'angle 0 à un angle de 90°
    delay(1000);

   servoMoteur.write(90);
   delay(1000);
   digitalWrite(ledPin, LOW);

   }
  else {

    digitalWrite(ledPin, HIGH);          //LED allumé au changement de couleur
    Serial.println();
    servoMoteur.write(0);                // Le servomoteur repasse en position 0
    delay(1000);

  }
}
```

# II. Phase de test

1. Ouverture du robinet et calibration des solutions avant et après l'équivalence avec la photorésistance

Les premiers tests d'ouvertures du robinets se sont montrés fructueux. On aurait pu craindre que les pales du servomoteur n'ait pas suffisamment de puissance pour faire tourner la clé de robinet, mais ce ne fut pas le cas.

Nous avons aussi réaliser un test pour calibrer la photorésistance lors du changement de couleur dans la boîte opaque. Après de multiples tests, nous avons remarqué que plusieurs paramètres pouvait influencer la valeur analogique de la photorésistance comme par exemple l'agitation, la position du bécher par rapport à la source lumineuse. De plus, nous avons remarqué que les gammes des valeurs ne sont pas stables entre deux expériences. Nous en avons conclu qu'avant chaque expérience, il était important de calibrer la solution d'HCl seul avant de commencer le titrage. Cependant lors du titrage, nous n'avons pas remarqué de différences significatives entre la solution d'HCl, et la solution de couleur rose à l'équivalence après ajout de la soude. Ces résultats peuvent être expliqués par la position de la photorésistance, du bécher et de la source lumineuse. En effet, les propriétés optiques du bécher jouant une influence non négligeable sur la lumière (

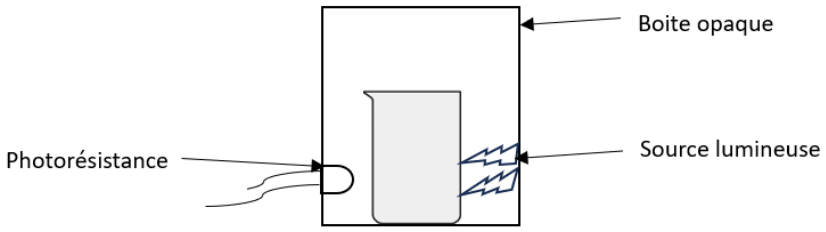


Figure 7: Schéma de la position du bécher, de la photorésistance et de la source de lumière

D'autres pistes ont alors été envisagés, comme par exemple placée la photorésistance au dessus de bécher. Cependant, cette piste à été abandonnée car la position de la source lumineuse étant situé tout en bas de la boite opaque, nous n'étions pas certain que la photorésistance puisse capter une intensité lumineuse suffisamment forte.  Autrement dit, les valeurs enregistrés par Arduino au niveau de la photorésistance aurait été très faible, ce qui aurait rendu difficile la détectabilité de la solution à l'équivalence (solution de couleur rose), et donc il aurait été plus compliqué d'arrêter le servomoteur lorsque l'on passe de la solution incolore à la solution de couleur rose. Au final, nous avons donc décider de remplacer la photorésistance par un capteur de couleur. Ce choix semble plus pertinent étant donné que notre dosage est basé sur le changement de couleur d'une solution à l'équivalence.

2. Utilisation du capteur de couleur

Le capteur de couleur TCS 34725 est un capteur qui détecte les couleur rouge, vert et bleu ainsi que le blanc. L'utilisation du capteur de couleur sera basé sur le même principe que la photorésistance sauf que cette fois-ci, nous n'utiliserons pas de boite opaque, et que le capteur de couleur sera placé au dessus du bécher. Nous avons donc modifié le code du titrage pour que celui-ci soit en adéquation avec le capteur de couleur et non plus avec la photorésistance.

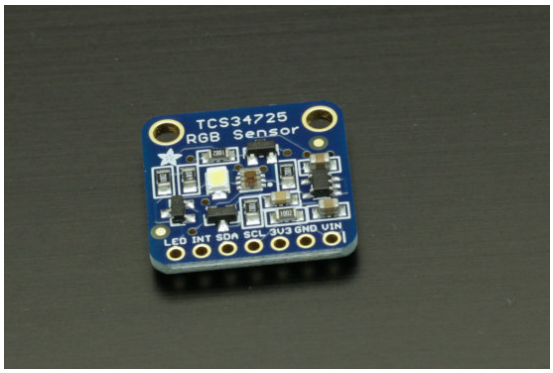


Figure 8: Capteur de couleur 34725 RGB[1]

```
#include <Wire.h>
#include "Adafruit_TCS34725.h"//on inclut le capteur de couleur dans la bibliothèque
#include <Servo.h>  // on inclut la bibliothèque servo

#define SERVO_STOP_ANGLE 0 // on définit l'angle initiales des pâles du servomoteur comme étant 0°
#define SERVO_START_ANGLE 90 // on définit l'angle de rotation des pâles du servomoteur comme étant de 90°
Servo servoMoteur;   // on crée un objet servo appelé servoMoteur


Adafruit_TCS34725 tcs = Adafruit_TCS34725 (TCS34725_INTEGRATIONTIME_50MS, TCS34725_GAIN_4X);



void setup() {
Serial.begin(9600);
servoMoteur.attach(9); // Servo moteur en position 9
servoMoteur.write(SERVO_START_ANGLE); // Position initial du servomoteur. Postion des pâles ont un angle de 0°
if (tcs.begin()) {
Serial.println("Found sensor"); // Si le capteur de couleur est connécté à Arduino alors Found Sensor s'affiche avant de lançer le titrage
 } else {
Serial.println ("No TCS 34725 found, check connections"); // Si le capteur de couleur n'est pas connécté à Arduino alors "No TCS 34725 found, check connections" s'affiche
while(1);
 }


}


void loop() {
```

```
  uint16_t clear, red, green, blue;

  tcs.getRawData(&red, &green, &blue, &clear);

  // Calculer la moyenne des valeurs de couleur pour réduire le bruit
  uint32_t sum = clear;
  uint16_t colorData[3] = {red, green, blue};
  for (int i = 0; i < 3; i++) {
    sum += colorData[i];
  }
  uint16_t average = sum / 4;

  Serial.print("Clear: "); Serial.print(clear);
  Serial.print(" Red: "); Serial.print(red);
  Serial.print(" Green: "); Serial.print(green);
  Serial.print(" Blue: "); Serial.println(blue);

  // Si la valeur de couleur claire (blanc) est situé en dessous de 450, alors le servomoteur s'arrête
"changement de couleur détecté" s'affiche sur l'écran du PC
  if (clear < 450) {
    Serial.println("Changement de couleur détecté!");
    servoMoteur.write(0);
    delay(1000);
  } else {   // Sinon le servomoteur tourne d'un angle de 0 à 90°. Il reste en position 0° pendant 5
secondes, puis en postion 90° pendant 2 secondes
      servoMoteur.write(0);
      delay(5000);

      servoMoteur.write(90);
      delay(2000);

    }

  }
```

3. Calibration du capteur de couleur

Comme pour la photorésistance, il fallu calibrer le capteur de couleur en fonction des deux solutions que nous avons, et déterminer la valeur affichés sur le PC par la couleur claire (le blanc). Le choix du blanc comme variable est totalement arbitraire. En effet, nous aurions tout aussi bien pu prendre les valeurs rouge, vert ou bleu. Comme pour la photorésistance, nous nous sommes rendus compte que l'agitation avait une influence sur les valeurs numériques enregistrés par le PC. Cela signifie que comme pour la photorésistance, la calibration des solutions doit se faire sous agitation. Après de multiples essais, nous avons pu déduire une gamme de valeur pour chacune des solutions:

- Pour la solution de HCl incolore, la couleur claire se situe aux alentours 489 selon les valeurs enregistrés sur le PC
- Pour le solution à l'équivalence (couleur rose), la couleur claire se situe aux alentours de 400 selon les valeurs enregistrés sur le PC.

Nous avons donc décidé dans le code que nous arrêterions le servomoteur dès que la couleur claire (clear) est inférieur à 450, sinon le servomoteur continue de tourner d'un angle de 0° vers un angle de 90°.

**Attention:** Nous constatons bien une différence significative entre la solution incolore et la solution rose. En revanche, il est important de préciser que les valeurs que nous avons peuvent changer entre deux expériences. Il est donc très important avant chaque titrage de calibrer la solution de HCl seul, d'attendre que cette valeur se stabilise, puis de commencer le titrage. Il faudra donc certainement changer la valeur seuil inscrit dans le code. Enfin, il est important de dire que pour calibrer la photorésistance on fixe la position du servomoteur à 0° dans le code.

Après deux tentatives nous avons réussi à automatiser un dosage colorimétrique sur Arduino en contrôlant l'ouverture du robinet de la burette et arrêtant de verser la solution titrante à l'équivalence.

# Bibliographie

[1]*Test Du Capteur de Couleur RGB TCS34725*. https://www.yoctopuce.com/FR/article/test-du-capteur-de-couleur-rgb-tcs34725. Accessed 27 Jan. 2024.

---

Revision #25
Created 7 December 2023 16:21:00 by Eleonore Victor
Updated 3 February 2024 10:58:02 by Al Khalil Ghida