Projet IETI:
Caméléon (Détection Automatique et Restitution de Couleurs)

Tony BRET
Logan LECONTE
Enzo SEBIANE
Abderrahmane SERI

April 2023

1 Introduction

Dans le cadre de notre projet du semestre 6, nous avons fait le choix de travailler sur un système de détection et de restitution de couleurs appelé DARC ou Caméléon. Notre projet repose donc sur la création d'un système de détection et de restitution des couleurs en reproduisant la principale caractéristique d'un caméléon. Notre système doit ainsi être capable de détecter une couleur puis de la reproduire à l'identique.
Ce système peut par exemple être utilisé dans le cadre d'un défilé de mode. Un mannequin porte alors un vêtement d'une couleur particulière qui sera analysée par notre système de detection.La restitution se fait alors en renvoyant la couleur à l'identique. Nous avons également réalisé un renvoir de la couleur complémentaire à celle détectée par notre système.

2 Cahier des charges et schéma fonctionnel

Nous avons entamé ce projet par la construction d'un cahier des charges à respecter dans le cadre d'utilisation du système que nous avons délimité. Ce cahier des charges est repris dans le tableau 1:

Critère de...	Explications
Couleur	-La couleur de sortie et la couleur d'entrée doivent être identiques. -Le caméléon doit être apte à renvoyer la couleur complémentaire à celle détectée en cas de besoin.
Puissance et énergie	Le caméléon peut reposer sur l'utilisation d'un bandeau de LEDs qui limite la puissance consommée (7W/m à pleine puissance).
Gestion du temps	-Les temps de détection et de réponse doivent être assez faibles (de l'ordre de 1s) pour assurer la continuité d'un défilé ou en cas de changement brusque de couleur. -Seules les LEDs entourant le mannequin s'allument à son passage et s'éteignent assez rapidement après.
Système	On doit pouvoir piloter les LEDs individuellement pour permettre la coordination entre le déplacement du mannequin et la gestion de la lumière.

Table 1: Cahier des charges du système

La figure 1 est un schéma fonctionnel du système et reprend les principales étapes de son fonctionnement.

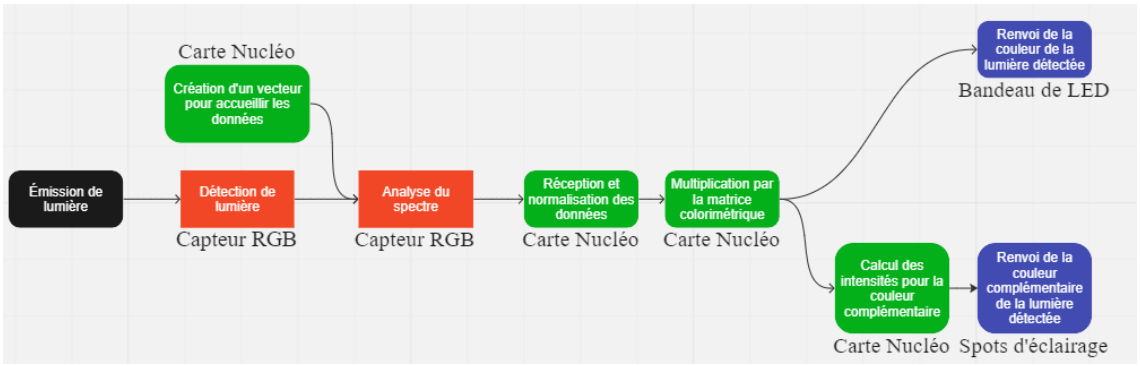
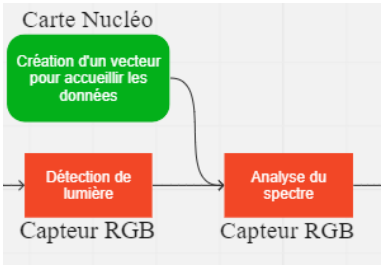


Figure 1: Schéma fonctionnel du système

3 Réalisation du projet

3.1 Le système de détection

3.1.1 Fonctionnement du capteur RGB



Le principal élément de notre système de détection est le capteur RGB. Le fonctionnement de ce capteur repose sur la détection de la lumière envoyée. Le capteur détermine alors les proportions de lumière Rouge, Verte et Bleue dans la lumière reçue. Il est d'ailleurs possible de lire ces quantités par l'intermédiaire du logiciel Tera Term. C'est ce que nous utiliseons notamment pour construire la matrice de passage entre le spectre du capteur et le spectre de l'oeil (voir partie suivante).

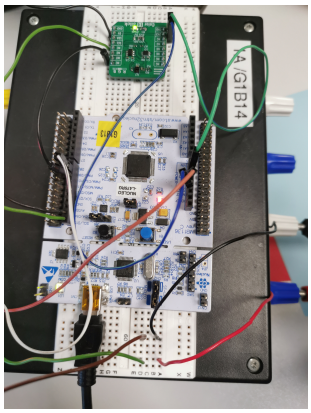
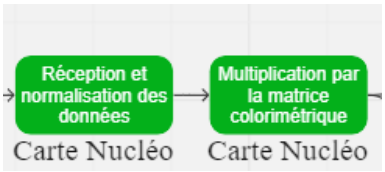


Figure 2: Système de détection avec capteur RGB

3.1.2 Normalisation et matrice de passage oeil/capteur-LED

Une étape importante pour le fonctionnement de notre capteur est d'effectuer une normalisation des valeurs. Nous avons en effet remarqué que les valeurs renvoyées par le capteur sur le logiciel TeraTerm étaient largement supérieures à 255, l'intensité maximale des LEDs, c'est pourquoi nous avons du réalisé une normalisation des valeurs à l'aide de la carte Nucléo et de la plateforme MBed avant d'envoyer les instructions au système de restitution. Cela est également utile pour réaliser notre matrice colorimétrique puisque sa création nécessite de travailler avec des valeurs entières entre 0 et 255.



Lors de la première séance de mise en commun (séance n°5), nous avons pu constaté que la couleur finale restituée par le bandeau de LEDs n'était pas totalement identique à celle qui était envoyée. Cela est en réalité dû au fait que les spectres lumineux de l'oeil et du capteur présentent des différences. Nous décidons donc de travailler avec les trois couleurs primaires de la synthèse additive pour déterminer ce que "voyait" le capteur à l'aide de Tera Term.
Considérons le vecteur colonne envoyé $V = (R\ G\ B)^t$ et la matrice M de passage entre le spectre de l'oeil et le spectre du capteur. Alors, le capteur envoie au bandeau le vecteur $U = MV$. On en déduit qu'il faut donc que la carte Nucléo multiplie ce dernier vecteur colonne par la matrice M^{-1} pour restituer correctement notre vecteur V.

3.2 Le système de restitution

Pour réaliser notre système de restitution, nous disposions d'un bandeau de 144 LEDs pilotables individuellement. Ce bandeau, alimenté par une tension continu de 5V, recevait et exécutait les commandes de la Carte Nucléo à laquelle il était lié. Chaque LED est constituée de 3 LED de couleur respectivement, rouge, verte et bleue, elle-même pilottables individuellement.
Pour commander ce bandeau nous importons les bibliothèques:
-Pixel Array: qui va permettre de créer le tableau contenant le nombre de LED à programmer.
-WS2812: qui permet de mettre à jour la matrice des LEDs et de régler la luminosité maximale que peut prendre chacune d'entre elles.

3.2.1 Restitution de la couleur détectée

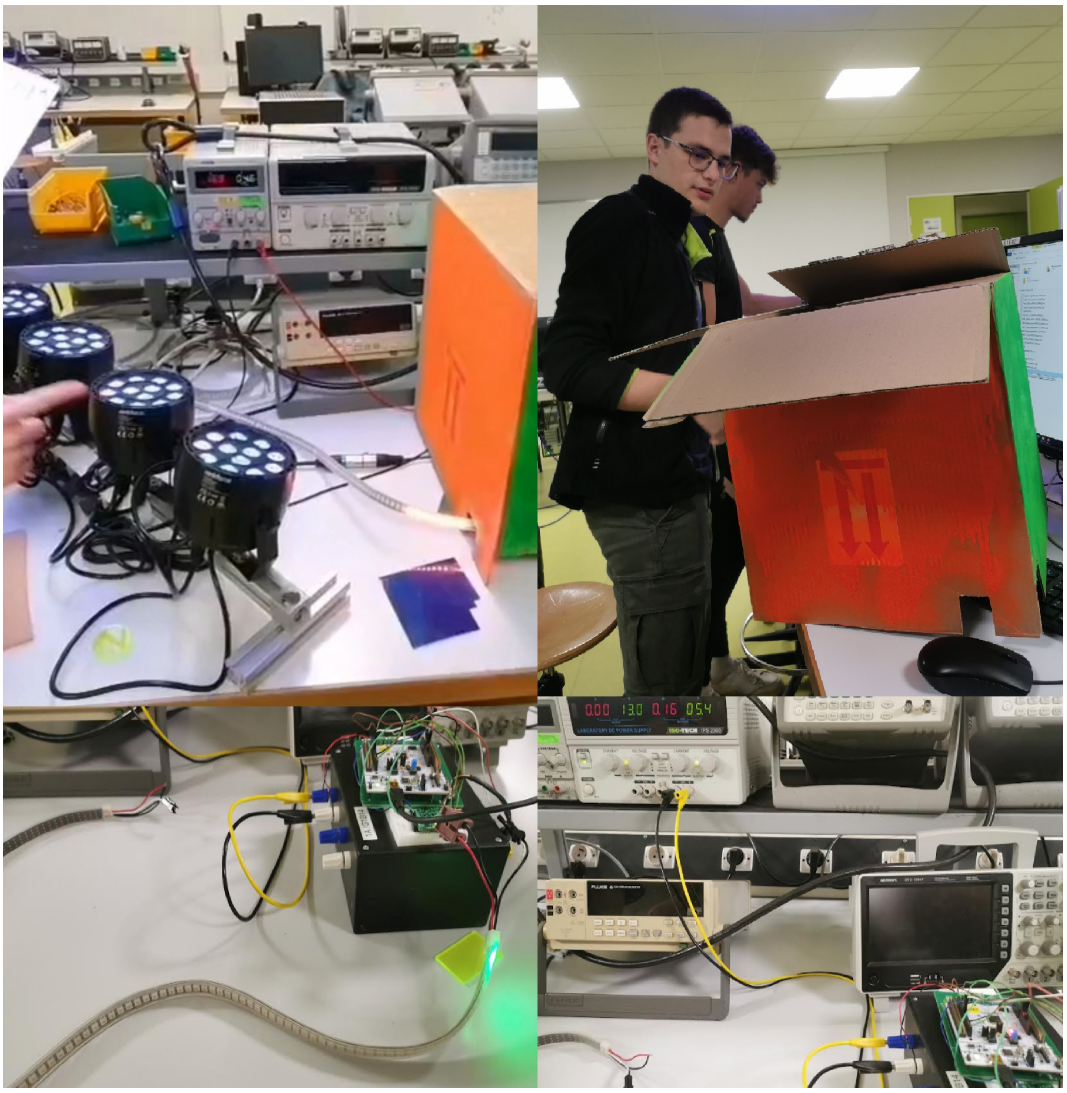
L'un des critères de notre cahier des charges est que le déclenchement du bandeau de LED se fasse de manière progressive. Le fait que les LEDs soient pilotables individuellement est donc un avantage considérable pour parvenir à remplir cette condition.
Il reste alors à choisir la vitesse à laquelle ces LEDs vont s'allumer. Ce dernier critère est déterminé par la vitesse à laquelle nous souhaitons faire avancer notre mannequin pour assurer une synchronisation et une continuité du défilé. Pour réaliser celà, nous avons imposé un temps d'attente entre l'allumage de deux LEDs successifs par l'intermédiaire d'un wait placé au sein de la boucle for servant à piloter individuellement les LED (voir code en annexe).
Par ailleurs, pour respecter ce critère de synchronisation, nous choisissons de n'allumer que le train de 10 LEDs parallèles au mannequin.
N.B.: Pendant la prise en main du bandeau de LEDs nous avons également réalisé un certain nombre d'essais (allumer toutes les LEDs avec une couleur aléatoire, implanter un ticker qui permet de répéter l'opération de détection/restitution même si toutes les LEDs n'ont pas eu le temps de s'allumer ou encore renvoyer deux couleurs en continus dans le cas où on enverrait deux mannequins à la fois). Cela nous a permis d'explorer plusieurs possibilités avant de choisir celle qui nous pariassait la plus intéressante dans ce que nous voulions réaliser.

3.2.2 Restitution de la couleur complémentaire

Dans un premier temps, nous envisagions de renvoyer la couleur complémentaire directement sur le bandeau de LED. Nous avons toutefois finalement fait le choix d'utiliser des spots pour effectuer cette action pour des raisons esthétiques (bien distinguer le système renvoyant la couleur détectée de celui renvoyant la couleur complémentaire.
Quant aux instructions transmises par la carte Nucléo, elles sont en réalité plutôt simple à coder. Il suffit en effet, pour chaque élement du vecteur RGB de retrancher la valeur détecter par le capteur à l'intensité maximale (par exemple 255-intensité de bleu détectée).

3.3 Mise en commun des deux systèmes et validation du prototype final

Après étude des systèmes de détection et de restitution, nous procédons à l'élaboration du prototype à présenter. Pour cela, nous avons fait le choix de construire une "boîte noire" où sera placé le système de détection. En effet, nous avons pu constaté lors de nos premiers essais que la lumière restituée par le bandeau de LEDs était toujours parasité malgré le fait que la matrice de passage ait été utilisée.
La conception et l'utilisation de la boïte noire permet d'éliminer la majorité de la lumière parasite reçue par le détecteur. Dans un premier temps, nous voulions aussi parvenir à éclairer l'intérieur de la boïte noire à l'aide d'un spot d'éclairage. Toutefois, nous avons constaté que la lumière émise par ce dernier était beaucoup trop puissante. Nous avons donc fait le choix d'utiliser une source lumineuse beaucoup moins étendue.
Nous avons alors choisi de tester les performances de notre système pour valider les critères du cahier des charges. La quasi-totalité de nos tests se basent sur des essais visuels (comparaison entre couleur de sortie et d'entrée, vérifier que les LED du bandeau s'allument à une vitesse ni trop lente ni trop rapide, que l'intensité renvoyée n'est pas trop élevée pour le confort visuel...). Nos multiples essais nous ont notamment permis de constaté certaines erreurs qui s'étaient glissées dans le calcul de notre matrice colorimétrique. Nous avons également du adapté certaines parties de notre code pour assurer la coordination entre les deux systèmes.



4 Bilan du projet

4.1 Planing d'avancement

Pour suivre l'avancement de notre projet, nous avons crée un dossier de partage sur google drive. Nous notions sur ce drive ce qui avait été effectué à chaque séance ainsi que les difficultés rencontrées, pour prendre le temps d'y réfléchir avant la séance suivante. Le tableau 2 reprend en détail les différentes étapes d'avancement de notre projet:

Date	Avancement du projet
06/02/2023	Choix du projet, discussion autour d'une application possible, du matériel et des idées à développer
13/02/2023	Élaboration des critères du cahier des charges, du système fonctionnel et répartition des tâches. Découverte d'un bandeau de plusieurs LEDs pilotables individuellement et tentative de programmation.
21/03/2023	Élaboration du système de détection: découverte du système RGB et de son mode de fonctionnement avec Tera Term. Élaboration du système de resitution: Prise en main du bandeau de LED, commande individuelle des LED, envoi d'une couleur et de sa couleur complémentaire.
28/03/2023	Expériences sur les bandeaux de LED: envoi de signaux aléatoires, implémentation d'un ticker pour réceptionner et modifier la couleur à intervalle régulier. Prise en main du capteur RGB: normalisation du signal obtenu. Prise en main de spots d'éclairage qui vont permettre d'éclairer la boîte noire et de trouver la couleur qu'il faudra renvoyer
04/04/2023	Mise en commun des deux systèmes. Construction et utilisation de la matrice de passage pour ajuster les erreurs dues au spectre du capteur Utilisation et programmation des spots pour envoyer la couleur complémentaire en ambiance de salle.
11/04/2023	Programmation des LEDs pour n'en allumer que 10. Pour assurer que la lumière reçue ne soit pas parasitée, discussion autour de la construction d'une boîte isolante. Début de la construction de la boîte.
18/04/2023	Poursuite de la réflexion sur l'utilisation de la boîte isolante et choix de la démonstration finale pour la présentation
09/05/2023	Séance Bilan et présentation du système.

Table 2: Tableau récapitulatif de l'avancement du projet

Concernant notre travail d'équipe il se divise en 3 parties. Tout d'abord les 2 premières séances ont été consacrées, en équipe entière, à la définition du système à mettre en place et des objectifs à atteindre. Pour les séances 3, 4 ainsi que la première moitié de la cinquième séance, nous avons privilégié un fonctionnement par binôme. Ainsi, Enzo et Logan se sont occupés de l'élaboration du système de détection tandis que Tony et Abderrahmane se sont attachés au fonctionnement du système de restitution. Enfin, les dernières séances ont été consacrées à la mise en commun des deux systèmes, aux corrections des éventuelles erreurs ainsi qu'à l'élaboration du prototype pour la démonstration finale.

4.2 Difficultés rencontrées

-Sur la partie détection: la matrice de passage doit être réalisé avec des sources lumineuses et des intensités bien précises qui ne doivent pas énormément varier entre elles ce qui limite nos possibilités d'éclairage.
-Sur la partie restitution

4.3 Compétences acquises

Au cours de ce projet nous avons eu l'occasion de développer:

-des comptéences techniques: améliorer nos compétences en langage C et avec MBED en particulier, différentes façons de piloter un bandeau de LED et un capteur RGB (par l'intermédiaire d'une boucle ou d'un ticker) permettant par exmple d'adapter nos systèmes selon les besoins de chacun(veut-on renvoyer des couleurs de manière périodique ou être apte à s'adapter à un changement brusque), analyser et traiter des données numériques sur TeraTerm selon les données et besoins du problème.

- des compétences de savoir-être: s'avoir s'adapter à une difficulté, un changement de situation, travailler en équipe, communiquer sur l'avancement du projet ou les difficultés rencontrées.

5 Annexes (code final)

```cpp
#include "mbed.h"
#include "PixelArray.h"
#include "WS2812.h"
#include <math.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define NB_LED  100 // Nombre de LEDs en cascade
#include "COLOR_14_CLICK.h"
#include    "DMX_MIDI.h"

UnbufferedSerial          debug_pc(USBTX, USBRX);
char        debugStr[64];
DigitalOut      debug_out(D13);
int i=0;
int NB_Spots=4;
/// Create a 4 integers array to collect R, G, B and IR data from the sensor
int             RGBir[4] = {0};
int maximum_LED = 0 ;

#define WAIT_TIME_MS 500
DigitalOut led1(LED1);

/// Create an I2C interface with specific pins : SDA / SCL
I2C                 my_color_i2c(D14, D15); // SDA / SCL
/// Create an interrupt input for the Color_14_Click Module INT pin
InterruptIn         my_color_int(D10);
/// Create a Color_14_Click module connection with an I2C interface and an Interrupt Input
Color_14_Click      my_sensor(&my_color_i2c, &my_color_int);

// Variables globales
PixelArray px(NB_LED);           // Tableau de pixels / Nb de LEDs en cascade
// For Nucleo F476 : 3, 12, 9, 12
WS2812 ws(D9, NB_LED, 3, 14, 9, 14);    // Lien vers la série de LEDs / Sortie utilisée-Nb de LEDs
// NE PAS MODIFIER LES 4 DERNIERS PARAMETRES




//Début de la fonction principale
int main()
{
    debug_pc.baud(115200);
    sprintf(debugStr, "Test\r\n");
    debug_pc.write(debugStr, strlen(debugStr));

    // Initialisation périphériques
    initDMX();
    //initMIDI();


    int k = 0;
    printf("Test.\n");
    /// PowerUp the module
    my_sensor.powerUp();
    /// Collect information about the sensor
    printf("Part ID = %d\r\n", my_sensor.getPartID());
    printf("Status = %d\r\n", my_sensor.getMainStatus());

```

```
    /// Initialize the sensor in RGB Mode
    my_sensor.initRGBSensor();
    /// Modify the analog gain of the sensor - 3x by default
    my_sensor.setGainRGB(COLOR_14_CLICK_LS_GAIN_6X);
    int i = 0;
    int m = 0;
    int j = 0;
    int n=0;
    int z=0;
    ws.useII(WS2812::GLOBAL);  // Initialisation de la liaison avec les LEDs
    ws.setII(0);               // Luminosité maximale à 0/255
    // Reinitialisation ecran
    for (i = 0; i < NB_LED; i++) {
        px.Set(i, 0);
    }
    ws.write(px.getBuf());      // Mise à jour de la matrice
    ws.setII(255);               // Luminosité maximale à 255
   /* toggle_led_ticker.attach(&toggle_led, 0.04);*/

    while (true)
    {
        k++;
        led1 = !led1;
        /// Collect R, G, B and IR data from the sensor
        my_sensor.readRGBIRValue(RGBir);

        //Multiplication du vecteur colonne (R,G,B) par l'inverse de la matrice passage
        RGBir[0]=0.003973612065868*RGBir[0]-0.000841644504455*RGBir[1]-0.00012960997242*RGBir[2];
        RGBir[1]=-0.000243192226733*RGBir[0]+0.004085048877599*RGBir[1]-0.00092897663777*RGBir[2];
        RGBir[2]=-0.00009547219487484*RGBir[0]-0.000472060469172*RGBir[1]+0.004046304708628*RGBir[2];

        // Recherche de l'intensité maximale
        if  (RGBir[0] > maximum_LED) maximum_LED=RGBir[0];
        if  (RGBir[1] > maximum_LED) maximum_LED=RGBir[1];
        if  (RGBir[2] > maximum_LED) maximum_LED=RGBir[2];

        // Normalisation de nos valeurs pour être entre 0 et 255
        RGBir[0]= (RGBir[0]*255)/maximum_LED;
        RGBir[1]= (RGBir[1]*255)/maximum_LED;
        RGBir[2]= (RGBir[2]*255)/maximum_LED;
        RGBir[3]= (RGBir[3]*255)/maximum_LED;
        printf("[%4d] R=%d / G=%d / B=%d / IR = %d \r\n\n ", k, RGBir[0], RGBir[1], RGBir[2], RGBir[3]); // affichage

        //Controle des spots
        for (i = 0; i < NB_Spots; i++) {   // valeurs pour spots 33 41 49 57
           dmx_data[8*i+33] = 0;    //mode (0=couleurs)
/*         dmx_data[8*i+34] = 10;  // selection couleur pour stobo / ex
           dmx_data[8*i+35] = 255; *///  vitesse strobo
           dmx_data[8*i+36] = 100;    //luminosité totale
           dmx_data[8*i+37] = 255-RGBir[0];    //rouge
           dmx_data[8*i+38] = 255-RGBir[1];    //vert
           dmx_data[8*i+39] = 255-RGBir[2];    //bleu
    }

```

```
115         /// Every 0.5s
116         thread_sleep_for(WAIT_TIME_MS);
117         updateDMX();
118         wait_us(1000);
119         for(i = 0; i < NB_LED; i++){
120             ws.setII(255);
121             ws.write(px.getBuf());
122             px.SetR(i,RGBir[0]);
123             px.SetG(i, RGBir[1]);
124             px.SetB(i, RGBir[2]);
125             wait_us(50000);     //temps d'attente entre le déclenchement de 2 LEDs successives
126             ws.write(px.getBuf());     //Mise à jour de la matrice à chaque passage dans la boucle
127             updateDMX();  // synchronisaion entre les spots et le bandeau
128 
129             // On ne souhaite allumer que 10 LEDs à la fois
130             if (i-10>=0){
131                 px.SetR(i,RGBir[0]);
132                 px.SetG(i, RGBir[1]);
133                 px.SetB(i, RGBir[2]);
134                 px.SetR(i-10,0);
135                 px.SetG(i-10,0);
136                 px.SetB(i-10,0);
137                 wait_us(50000);
138             }
139             //On éteint toutes les LEDs après passage de la voiture/mannequin
140             if(i==NB_LED-1){
141                 for (z=0; z<NB_LED; z++){
142                     px.SetR(z,0);
143                     px.SetG(z,0);
144                     px.SetB(z,0);
145                 }
146             }
147 
148         }
149     }
150 }
```