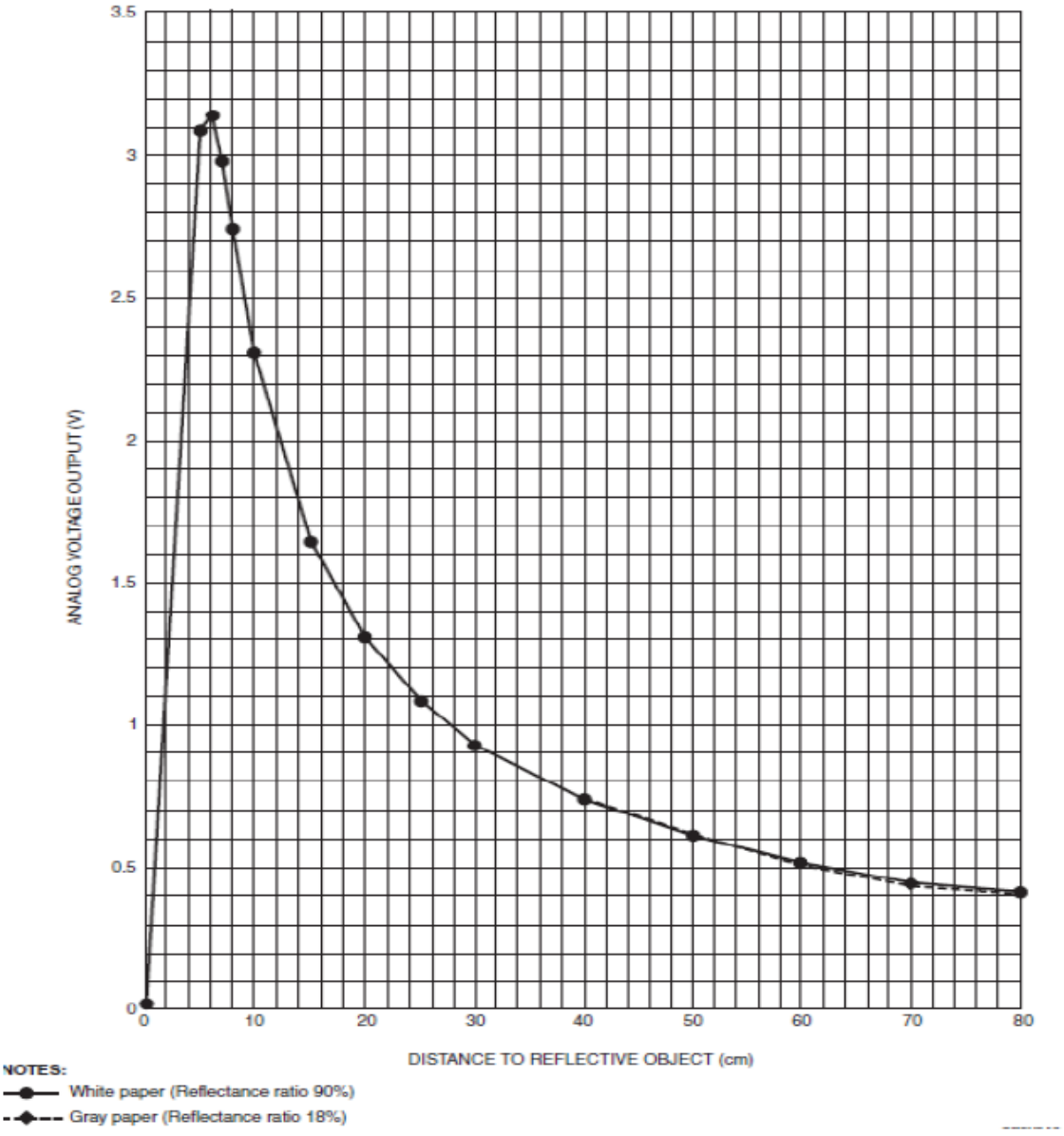
TELEMETRE INFRAROUGE

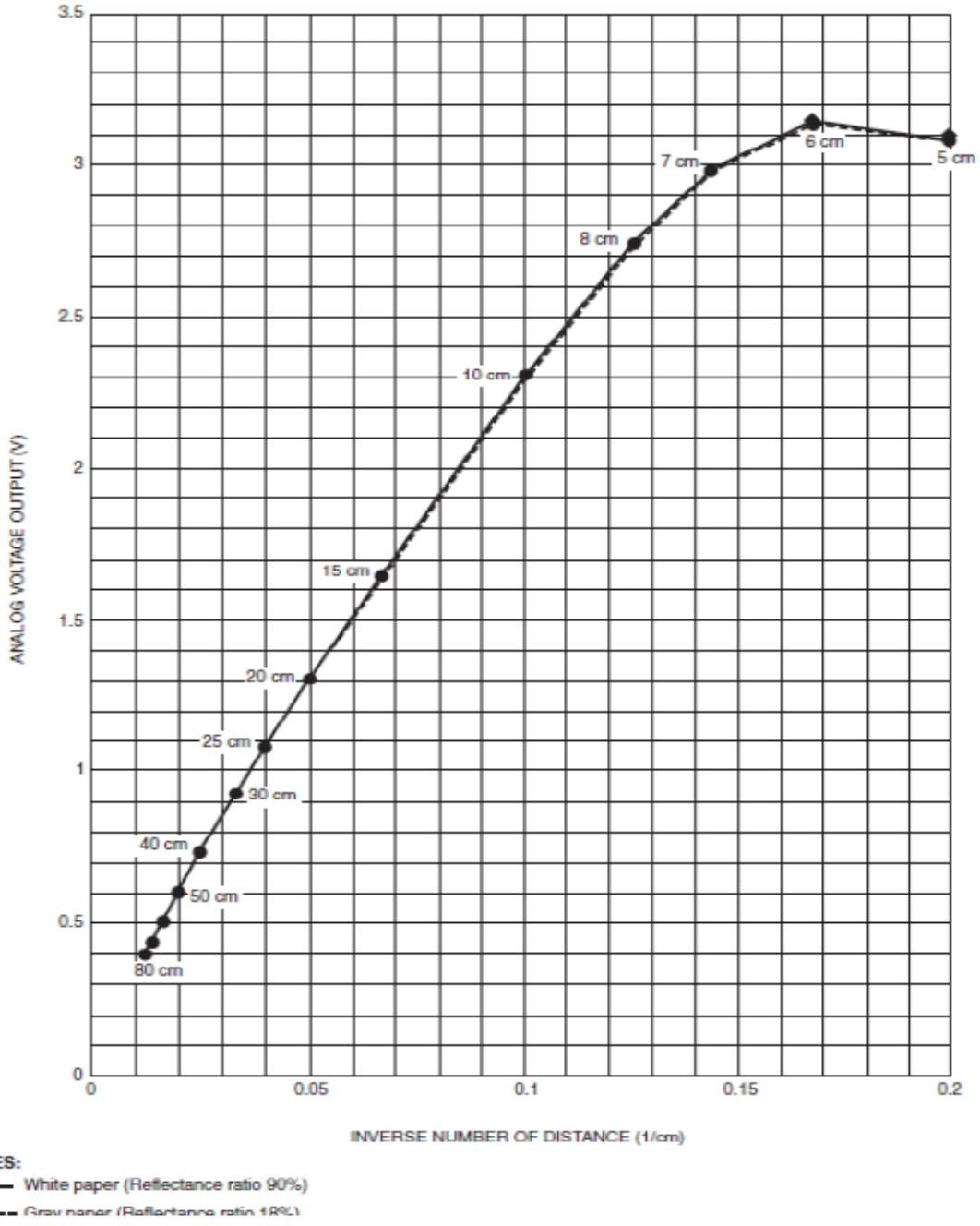
Esteban Chevalier-Drevon
Matyas Ricci
Groupe F

Le télémètre infrarouge

1. Analyse de courbe :

On souhaite tracer sur EXCEL la courbe Tension=f(distance) de notre télémètre infrarouge.
Les courbes tirées de la documentation technique (GP2Y0A21YK) sont les suivantes :





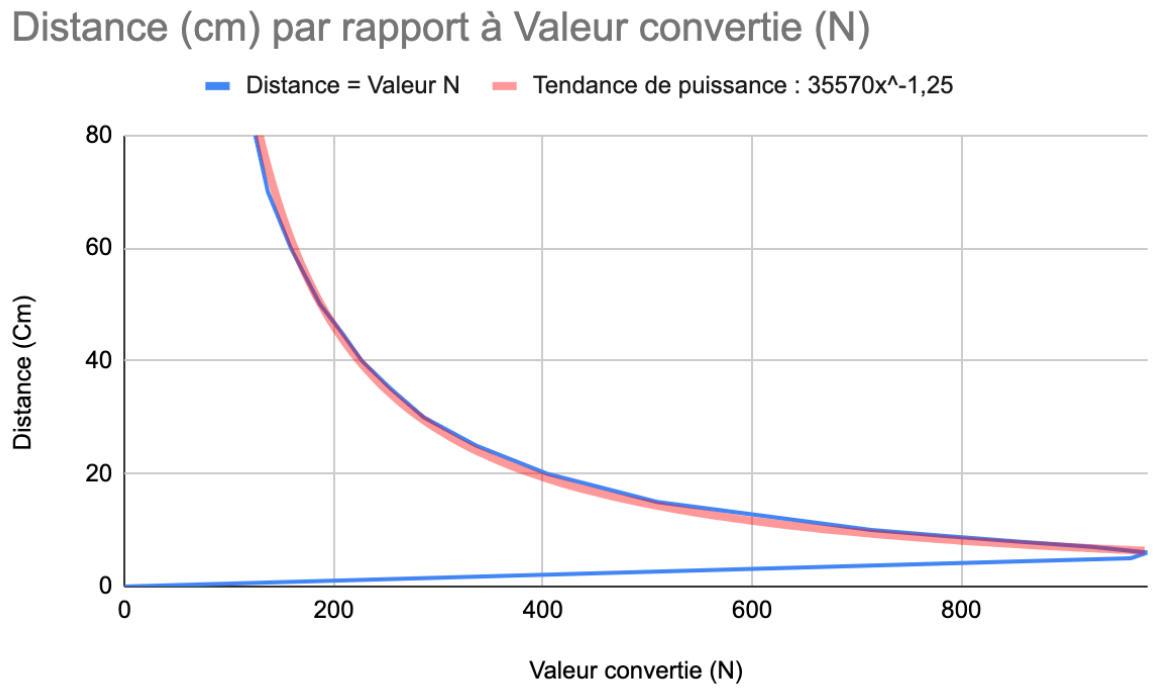
Relevez les valeurs sur la courbe et remplir le tableau suivant :

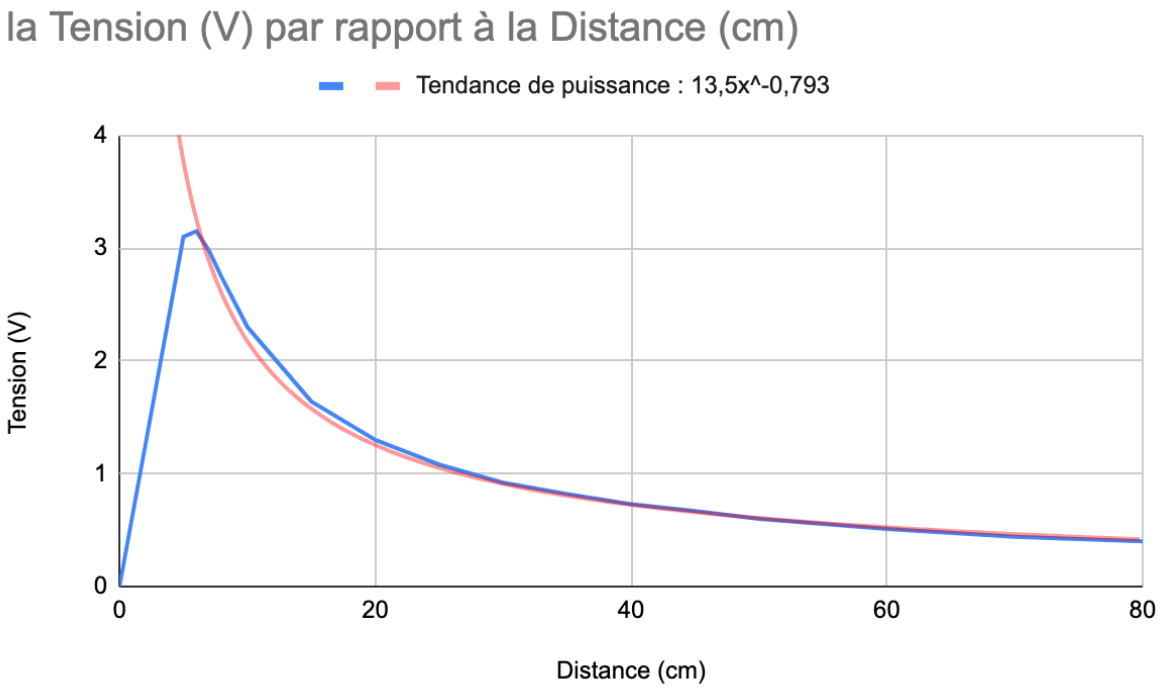
Distance (cm)	0	5	6	7	8	9	10	15	20
Tension (V)	0	3,1	3,15	2,98	2,74	2,52	2,3	1,64	1,3
Valeur Convertie N	0	635	645	610	561	516	471	336	266

Distance (cm)	25	30	35	40	45	50	60	70	80
Tension (V)	1,08	0,92	0,82	0,73	0,67	0,6	0,51	0,44	0,4
Valeur Convertie N	221	188	168	150	137	123	105	90	82

Tracer les courbes Tension=f(Distance) et Distance=f (Valeur convertie N) sous OpenOffice.

Tracer la courbe de tendance puissance de la fonction Distance D=f (valeur convertie N). Noter l'équation fournie par le logiciel.





2. Analyse de documentation

A l'aide de la documentation (Infrarouge GP2Y0A21YK0F), extraire les informations suivantes :

Alimentation ? 4,5V – 5,5V
Consommation ? Consommation typique de 30mA
Distance de détection ? De 10 cm à 80 cm
Type de sortie ? Analogique (de type tension)

GP2Y0A21YK0F

Distance Measuring Sensor Unit
Measuring distance: 10 to 80 cm
Analog output type

■Description

GP2Y0A21YK0F is a distance measuring sensor unit, composed of an integrated combination of PSD (position sensitive detector) , IRED (infrared emitting diode) and signal processing circuit.
The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method.
This device outputs the voltage corresponding to the detection distance. So this sensor can also be used as a proximity sensor.

■Features

1. Distance measuring range : 10 to 80 cm
2. Analog output type
3. Package size : 29.5×13×13.5 mm
4. Consumption current : Typ. 30 mA
5. Supply voltage : 4.5 to 5.5 V

■Agency approvals/Compliance

1. Compliant with RoHS directive (2002/95/EC)

■Applications

1. Touch-less switch
   (Sanitary equipment, Control of illumination, etc.)
2. Robot cleaner
3. Sensor for energy saving
   (ATM, Copier, Vending machine)
4. Amusement equipment
   (Robot, Arcade game machine)

3. Programme 1 : Acquisition et affichage des valeurs sur le LCD

Ecrire un programme qui détecte et affiche la distance de l'objet sur le moniteur série et l'écran LCD.
Quelle valeur est affichée s'il n'y a aucun obstacle ?

Lorsqu'il n'y a aucun obstacle, la valeur affichée est élevée (+ de 50 cm) et cette valeur fluctue.

Programme :

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2); // Adressage de notre écran LCD à
l'adresse 0x27

#define IrG A2
#define IrD A6


void setup() {
    pinMode(IrD, INPUT);
    pinMode(IRG, INPUT);
    Serial.begin(9600);
    lcd.init();
    lcd.backlight();
    analogReference(EXTERNAL);

}

void loop() {
    int DistanceD, DistanceG;
    DistanceD = analogRead(IrD);
    DistanceG = analogRead(IrG);
    Serial.println(DistanceG);
    lcd.setCursor(0,0);
    lcd.print((String)DistanceG  + "   " + DistanceD + " ");
    lcd.setCursor(0,1);
    lcd.print(pow(DistanceG, - 1.25) * 49570);
    lcd.print(" ");
    lcd.print(pow(DistanceD, - 1.25) * 49570);
    lcd.print(" ");
    delay(1500);
}
```

4. Test d'intégration N°1 :

Faire avancer le robot et le faire s'arrêter lorsqu'il est à 20 cm d'un obstacle.

Test réalisé et validé par l'enseignant

Programme :

```
#define In1 2
#define In2 3
#define In3 4
#define In4 7
#define EnA 5
#define EnB 6

#define IrG A2
#define IrD A6
#define IrF A7

void cmd_motor(bool motor, bool direction = 0, short power = 0);


void setup() {
    Serial.begin(9600);
    pinMode(In1, OUTPUT);
    pinMode(In2, OUTPUT);
    pinMode(In3, OUTPUT);
    pinMode(In4, OUTPUT);
    pinMode(EnA, OUTPUT);
    pinMode(EnB, OUTPUT);
    pinMode(IrD, INPUT);
    pinMode(IRG, INPUT);
    lcd.init();
    lcd.backlight();
    analogReference(EXTERNAL);

}

void loop() {
    static unsigned long int t0;
    static unsigned short DistanceG, DistanceD, DistanceF;
    if (millis() - t0 > 25) {
        t0 = millis();
        DistanceD = (pow(analogRead(IrD), - 1.25) * 49570);
        DistanceG = (pow(analogRead(IrG), - 1.25) * 49570);
        DistanceF = (pow(analogRead(IrF), - 1.25) * 49570);

        if (DistanceD >= 80) DistanceD = 70;
        if (DistanceG >= 80) DistanceG = 70;
```

```
        if (DistanceF >= 80) DistanceF = 70;
        if (DistanceF > 20 && DistanceD > 20 && DistanceG > 20) {
            cmd_motor(0, 1, 50);
            cmd_motor(1, 1, 50);
        } else {
            cmd_motor(0, 1, 0);
            cmd_motor(1, 1, 0);
        }
    }
}

void cmd_motor(bool motor, bool direction = 0, short power = 0) {
    float kfactor = 1;
    if (motor) {
        if (power == 0) {
            analogWrite(EnB, 0);
        }
        power = (power * 2.55) * kfactor;
        if (direction) {
            digitalWrite(In1, LOW);
            digitalWrite(In2, HIGH);
        } else {
            digitalWrite(In2, LOW);
            digitalWrite(In1, HIGH);
        }
        analogWrite(EnB, power);
    } else {
        if (power == 0) {
            analogWrite(EnA, 0);
        }
        power = (power * 2.55);
        if (direction) {
            digitalWrite(In3, HIGH);
            digitalWrite(In4, LOW);
        } else { // Marche arrière
            digitalWrite(In3, LOW);
            digitalWrite(In4, HIGH);
        }
        analogWrite(EnA, power);
    }
}
```

5. Test d'intégration N°2 :

Eviter pendant une minute les obstacles dans une enceinte définie par l'enseignant.

Test réalisé et validé par l'enseignant

Programme :

```
#define In1 2
#define In2 3
#define In3 4
#define In4 7
#define EnA 5
#define EnB 6

#define IrG A2
#define IrD A6
#define IrF A7

void cmd_motor(bool motor, bool direction = 0, short power = 0);


void setup() {
    Serial.begin(9600);
    pinMode(In1, OUTPUT);
    pinMode(In2, OUTPUT);
    pinMode(In3, OUTPUT);
    pinMode(In4, OUTPUT);
    pinMode(EnA, OUTPUT);
    pinMode(EnB, OUTPUT);
    pinMode(IrD, INPUT);
    pinMode(IRG, INPUT);
    lcd.init();
    lcd.backlight();
    analogReference(EXTERNAL);

}

void loop() {
    static unsigned long int t0, t1, inLoopTimer
    static unsigned short DistanceG, DistanceD, DistanceF;
    if (millis() - t0 > 25) {
        t0 = millis();
        if (t0 > 60 * 1000) return;
        DistanceD = (pow(analogRead(IrD), - 1.25) * 49570);
        DistanceG = (pow(analogRead(IrG), - 1.25) * 49570);
        DistanceF = (pow(analogRead(IrF), - 1.25) * 49570);

        if (DistanceD >= 80) DistanceD = 70;
```

```
        if (DistanceG >= 80) DistanceG = 70;
        if (DistanceF >= 80) DistanceF = 70;

        if (DistanceF < 15 || DistanceD < 15 || DistanceG < 15) {
            inLoopTimer = millis();

            while (DistanceF < 20 || DistanceD < 20 || DistanceG < 20) {
                if (millis() - t1 > 10) {
                    t1 = millis();

                    if (millis() - inLoopTimer > 2000 || inLoop > 400) {
                        inLoopTimer = millis();
                        inLoop = 0;
                        cmd_motor(1, 0, 80);
                        cmd_motor(0, 0, 80);
                        delay(100);
                        cmd_motor(1, 1, 0);
                        cmd_motor(0, 1, 0);
                        return;
                    }

                    if (DistanceD > DistanceG) {
                        cmd_motor(1, 0, 35);
                        cmd_motor(0, 0, 25);
                    }
                    else {
                        cmd_motor(1, 0, 25);
                        cmd_motor(0, 0, 35);
                    }

                    DistanceD = (pow(analogRead(IrD), -1.25) * 49570);
                    DistanceG = (pow(analogRead(IrG), -1.25) * 49570);
                    DistanceF = (pow(analogRead(IrF), -1.25) * 49570);

                    if (DistanceD >= 80) DistanceD = 70;
                    if (DistanceG >= 80) DistanceG = 70;
                    if (DistanceF >= 80) DistanceF = 70;
                }
                return;
            }
        }
    }
}

void cmd_motor(bool motor, bool direction = 0, short power = 0) {
```

```
    float kfactor = 1;
    if (motor) {
        if (power == 0) {
            analogWrite(EnB, 0);
        }
        power = (power * 2.55) * kfactor;
        if (direction) {
            digitalWrite(In1, LOW);
            digitalWrite(In2, HIGH);
        } else {
            digitalWrite(In2, LOW);
            digitalWrite(In1, HIGH);
        }
        analogWrite(EnB, power);
    } else {
        if (power == 0) {
            analogWrite(EnA, 0);
        }
        power = (power * 2.55);
        if (direction) {
            digitalWrite(In3, HIGH);
            digitalWrite(In4, LOW);
        } else { // Marche arrière
            digitalWrite(In3, LOW);
            digitalWrite(In4, HIGH);
        }
        analogWrite(EnA, power);
    }
}
```