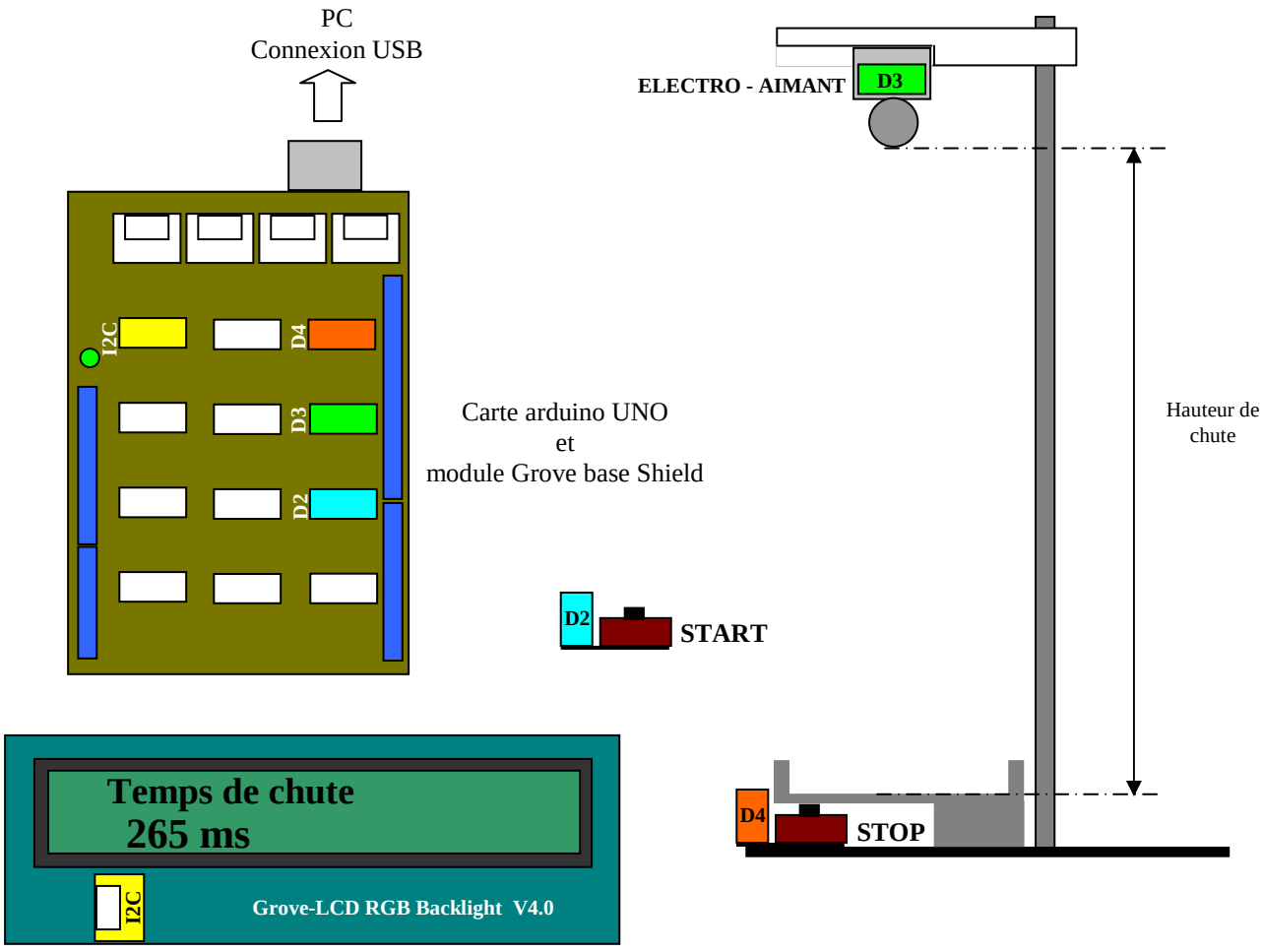
ETUDE DE LA CHUTE LIBRE

PROGRAMMATION
MBLOCK ET ARDUINO

P.Langlois Mars 2019 / G.Eiffel
Pascal.langlois@ac-dijon.fr

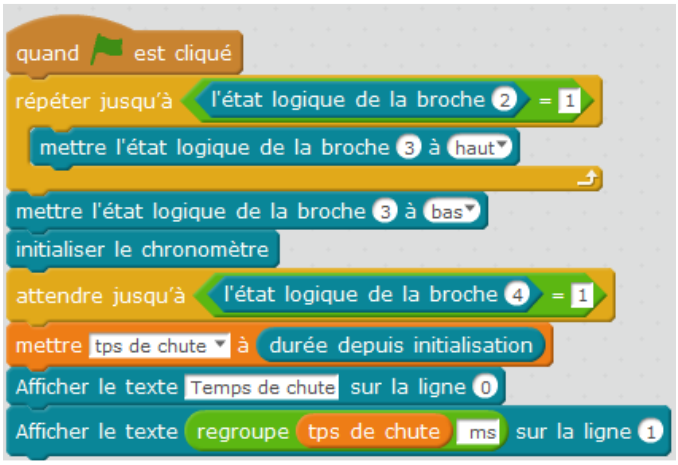
I. Mesure du temps de chute d'une bille :

1.1 Montage :



P.Langlois Mars 2019 / G.Eiffel
Pascal.langlois@ac-dijon.fr

1.2 Programme mBlock



// Electro aimant collé jusqu'à START = 1

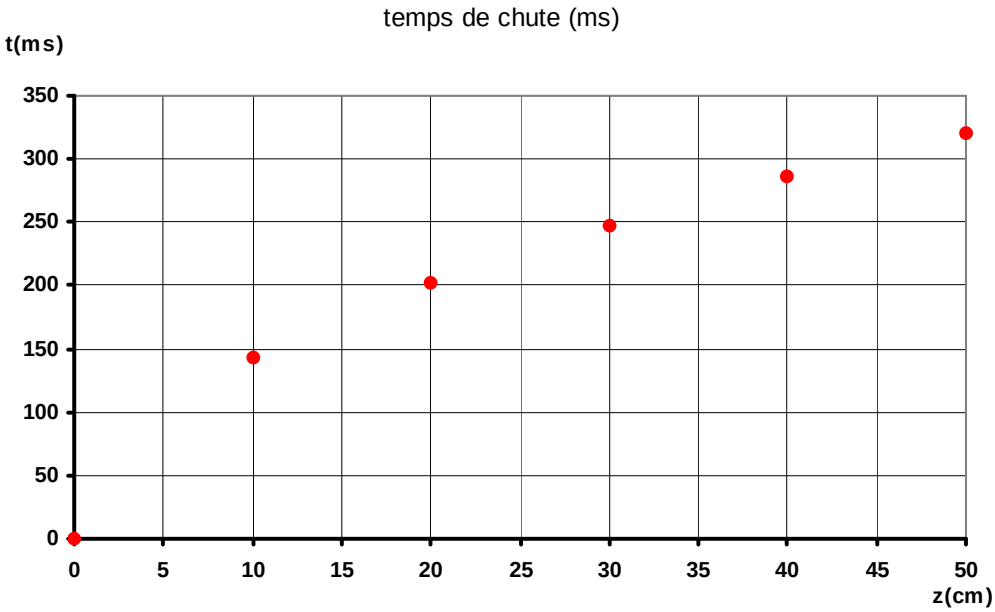
// Lâcher de la bille et initialisation du chronomètre

// Lecture du chronomètre pour STOP = 1

// Affichage du résultat en ms

1.3 Mesures :

Hauteur de chute (cm)	temps de chute (ms)
0	0
10	143
20	202
30	248
40	286
50	320



On observe un décalage de quelques ms entre la valeur mesurée et la valeur théorique principalement dû au temps de lâcher de la bille.

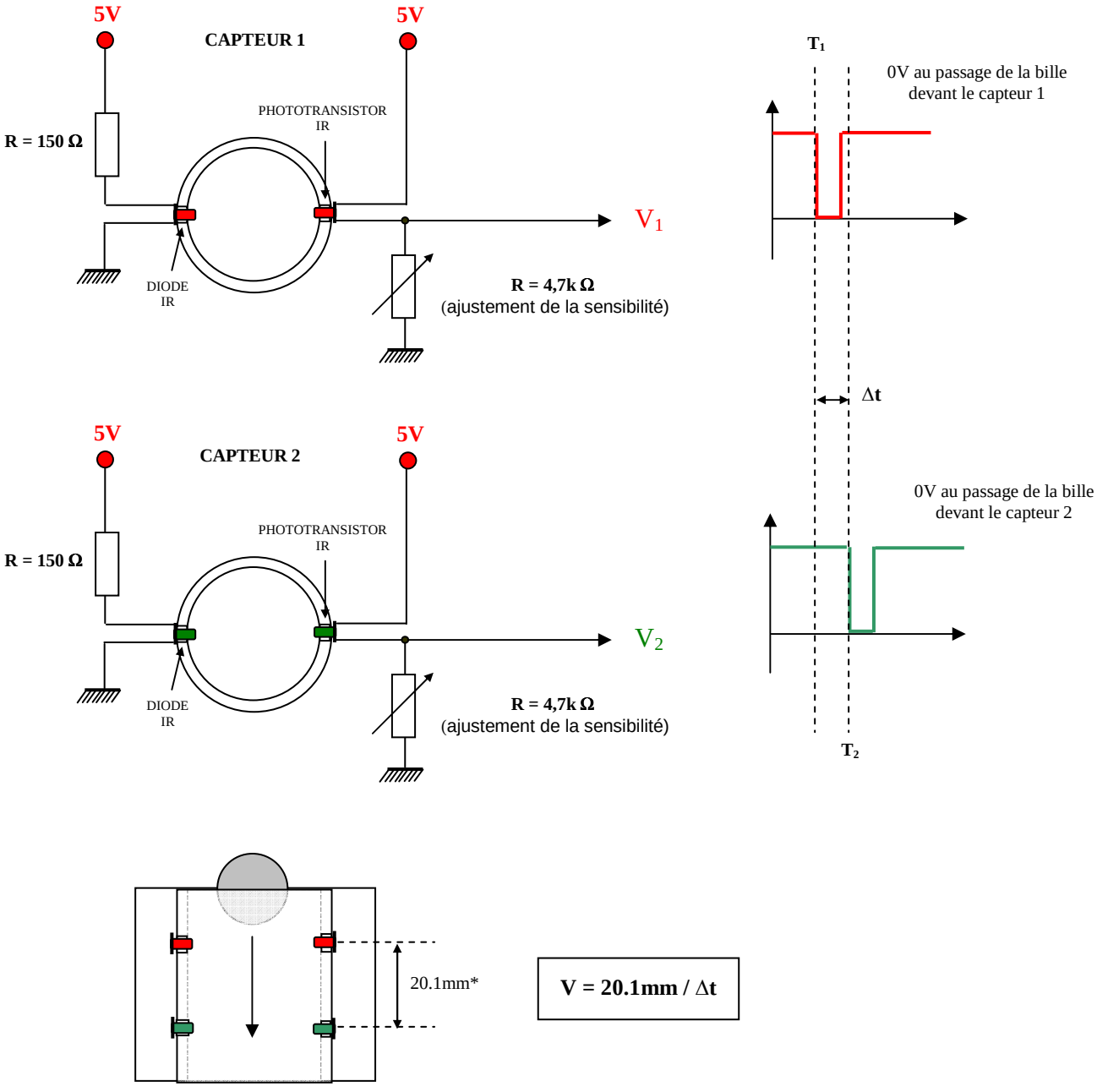
II. Mesure de la vitesse d'une bille en chute libre :

2.1 Mise en œuvre d'un capteur de vitesse

1.11 principe :

Un dispositif avec leds et phototransistors infrarouges permet de mesurer les temps de passage de la bille devant deux capteurs.

1.12 Dispositif :

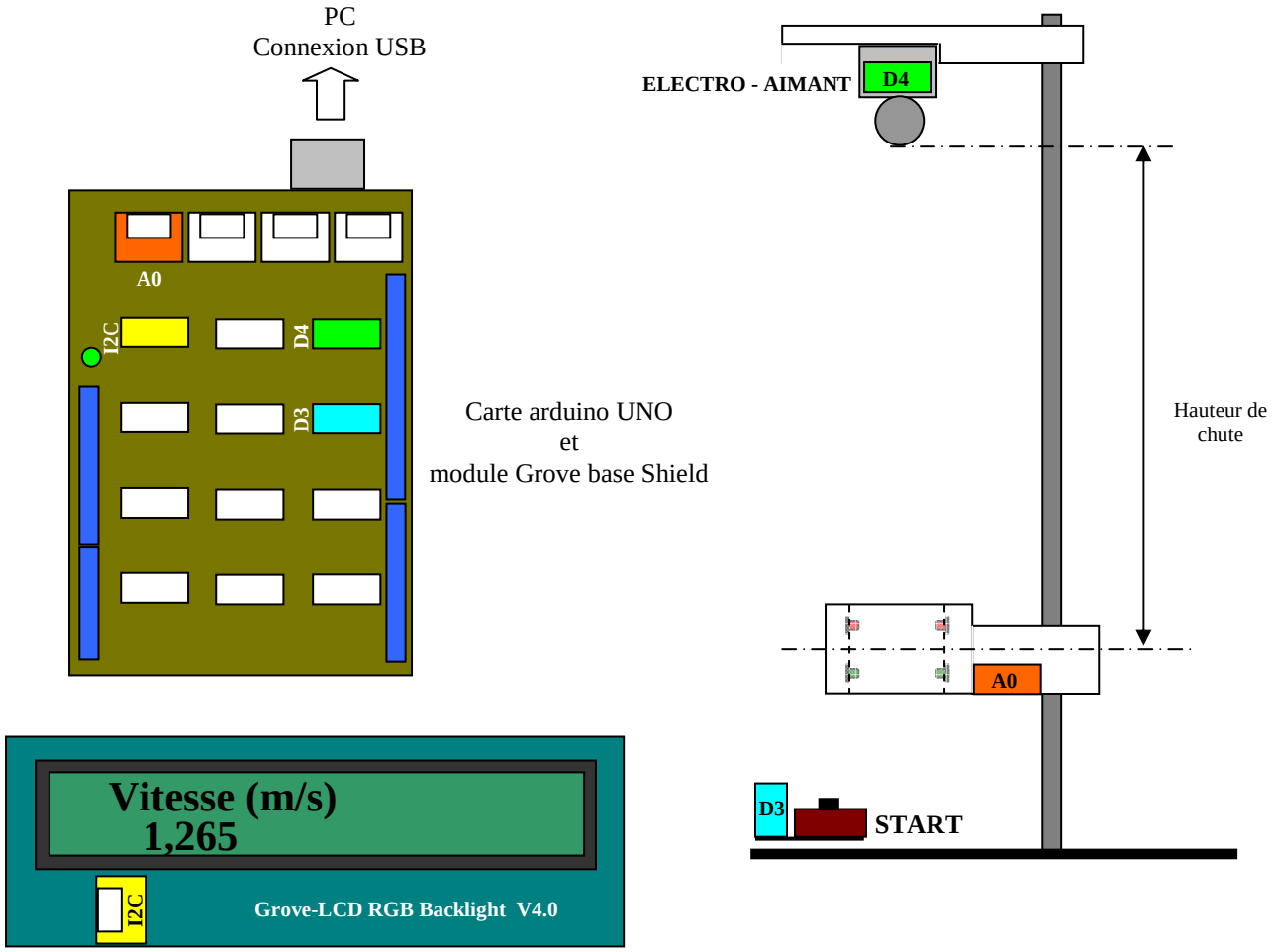


V = 20.1mm / Δt

* l'écartement entre les deux capteurs doit être mesuré très précisément (ici 20.1mm).

P.Langlois Mars 2019 / G.Eiffel
Pascal.langlois@ac-dijon.fr

2.2 Montage :



2.3 Programme arduino :

```
#include <Wire.h>       // Bibliothèque permettant  de communiquer avec les composants utilisant le protocole I2C
#include <rgb_lcd.h>    // Bibliothèque permettant la prise en charge de l'afficheur LCD RGB

rgb_lcd lcd;
const int colorR = 255; // Déclaration de l'afficheur LCD RGB avec la valeur pour les trois couleurs RGB
const int colorG = 255;
const int colorB = 255;

const byte pinEA = 4;         // déclaration des broches correspondant aux
const byte pinSTART = 3;      // connexions de l'électro aimant et du bouton Start

                              // declaration des variables
int VDHref;                   // Signal de sortie du capteur 1 sans présence de la bille
int VDBref;                   // Signal de sortie du capteur 2 sans présence de la bille
int VDH;                      // Signal de sortie du capteur 1
int VDB;                      // Signal de sortie du capteur 2
unsigned long T1;             // date de passage devant le capteur 1
unsigned long T2;             // date de passage devant le capteur 2
float DT;                     // temps écoulé entre T1 et T2
```


P.Langlois Mars 2019 / G.Eiffel
Pascal.langlois@ac-dijon.fr

```
float V;                    // vitesse de la bille

void setup() {              // Programme qui s'exécute au démarrage.

lcd.begin (16,2);           // texte sur les deux lignes de l'afficheur LCD (deux lignes de seize caractères)
lcd.setRGB (colorR, colorG, colorB);


pinMode (pinEA,OUTPUT);     // déclaration de la broche de l'électro-aimant en sortie
pinMode (pinSTART,INPUT);   // déclaration de la broche du bouton Start en entrée

digitalWrite (pinEA,HIGH);  // Mise à l'état haut (5V) de la sortie EA (aimantation de l'électro-aimant)

T1 = 0;                     // Initialisation de T1 et T2
T2 = 0;

VDHref = analogRead (A0);   // mesure des signaux de sortie du capteur sans la présence de la bille
VDBref = analogRead (A1);   // ces signaux sont lus sur les entrées analogiques A0 et A1.

Serial.begin(9600);         // débit de communication série
Serial.println(VDHref);     // affichage de VDHref et VDBref via le moniteur série d'arduino.
Serial.println(VDBref);

lcd.setCursor (1,0);        // afficher « vitesse (m/s) » sur la ligne 1 de l'afficheur
lcd.print ("vitesse (m/s)");


void loop() {               // programme qui s'exécute en boucle

VDH = analogRead (A0);      // lecture des signaux du capteur
VDB = analogRead (A1);


while(digitalRead(pinSTART)==LOW);  // la sortie EA passera à 0 (0V) que lorsque le bouton poussoir
digitalWrite (pinEA,LOW);           // passera à 1 (5V)


  while((VDHref/2) < (analogRead (A0)));  // Lorsque la mesure de A0 est inferieure à VDHref/2, T1 prend la
T1 = micros();                            // valeur de micros. Micros ( ) Renvoie le nombre de
                                          // microsecondes depuis le démarrage du programme.


while((VDBref/2) < (analogRead (A1)));  // Lorsque la mesure de A1 est inferieure à VDBref/2, T2 prend la
T2 = micros();                          // valeur de micros.


DT = (float)(T2 - T1)/1000.00;          //  Calcul de Δt en ms avec virgule

V = 0.0201*(1000/DT);                   // Calcul de la vitesse en m/s (la mesure très précise de l'écartement
                                        // entre les deux capteurs donne 20 mm et 100µm)

  Serial.print("T1 = ");Serial.println(T1);
  Serial.print("T2 = ");Serial.println(T2);  // affichage de T1, T2 et DT via le moniteur série d'arduino
  Serial.print("DT = ");Serial.println(DT);
lcd.setCursor (0,1);                    // Affichage sur la ligne 2 de l'afficheur de v avec 3 chiffres après la
```

```
lcd.print (V,3);                    // virgule

if ((T1 != 0)&&(T2 != 0)){          // Si T1 et T2 ≠ 0 (acquisition terminée), réinitialisation de T1 et T2 et
                                    // remise de la broche de l'EA à 1. Prêt pour une nouvelle acquisition.
  T1 = 0;
  T2 = 0;
  delay(500);
  digitalWrite (pinEA,HIGH);
 }
 }
```

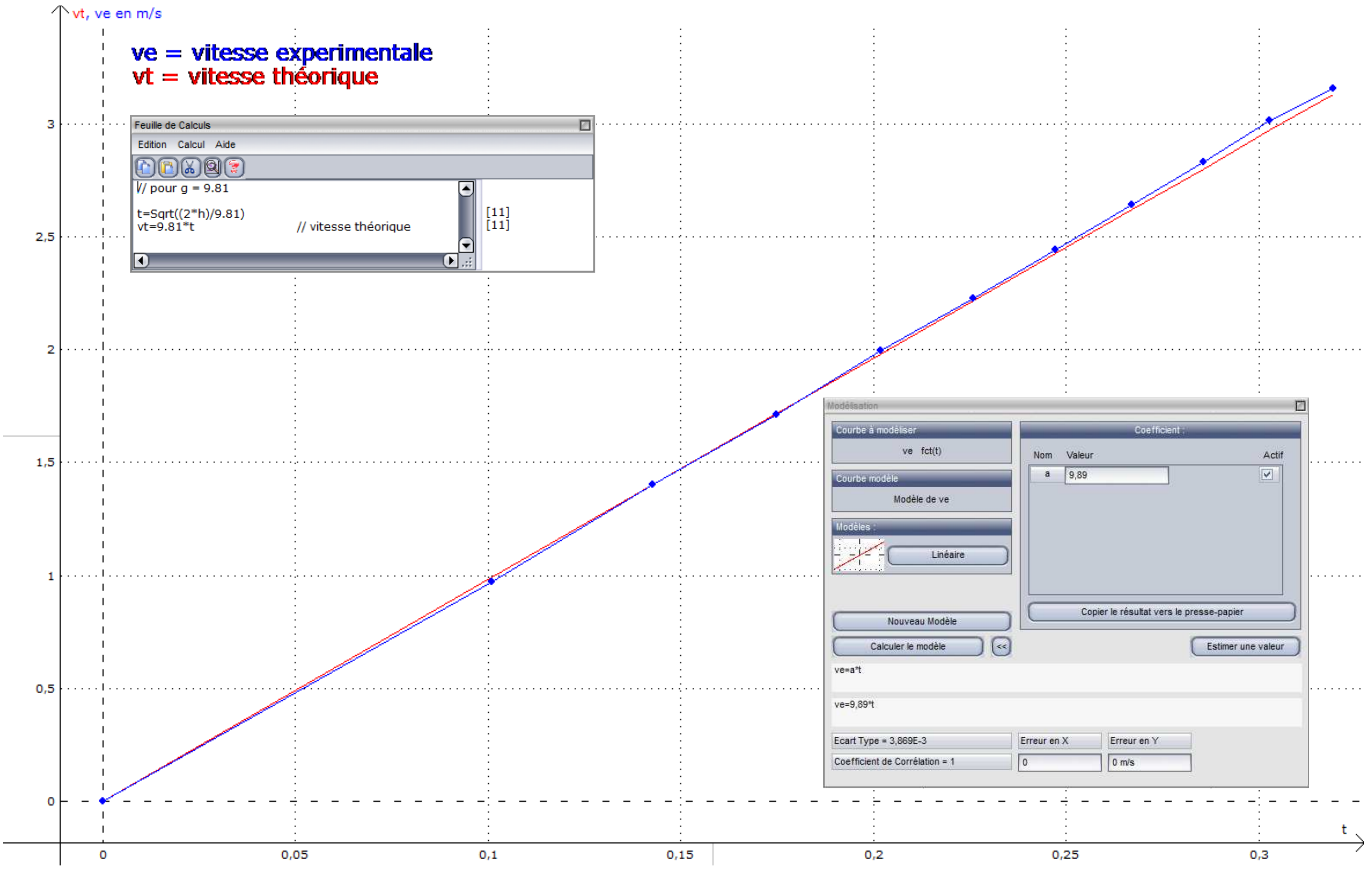
2.4 Mesures:

z(m)	10 mesures de v (m/s)										**v(m/s)***
0,05	0,974	0,974	0,974	0,974	0,974	0,974	0,974	0,974	0,974	0,974	**0,974**
0,1	1,401	1,401	1,401	1,401	1,401	1,401	1,401	1,401	1,401	1,401	**1,401**
0,15	1,702	1,718	1,718	1,718	1,718	1,718	1,718	1,702	1,718	1,718	**1,714**
0,2	2,002	2,002	2,002	1,98	2,002	2,002	2,002	2,002	2,002	2,002	**1,999**
0,25	2,249	2,221	2,221	2,221	2,221	2,221	2,249	2,221	2,221	2,221	**2,226**
0,3	2,429	2,461	2,429	2,461	2,429	2,461	2,461	2,429	2,429	2,461	**2,445**
0,35	2,64	2,679	2,64	2,64	2,64	2,64	2,64	2,64	2,64	2,64	**2,643**
0,4	2,846	2,846	2,802	2,846	2,846	2,802	2,846	2,846	2,802	2,846	**2,832**
0,45	3,036	2,986	3,036	2,986	3,036	2,986	3,036	2,986	3,036	3,036	**3,016**
0,5	3,14	3,14	3,195	3,14	3,14	3,195	3,14	3,14	3,195	3,14	**3,156**

* valeur moyenne des 10 mesures.

2.5 Courbes.

Vitesse (théorique et expérimentale) en fonction du temps.

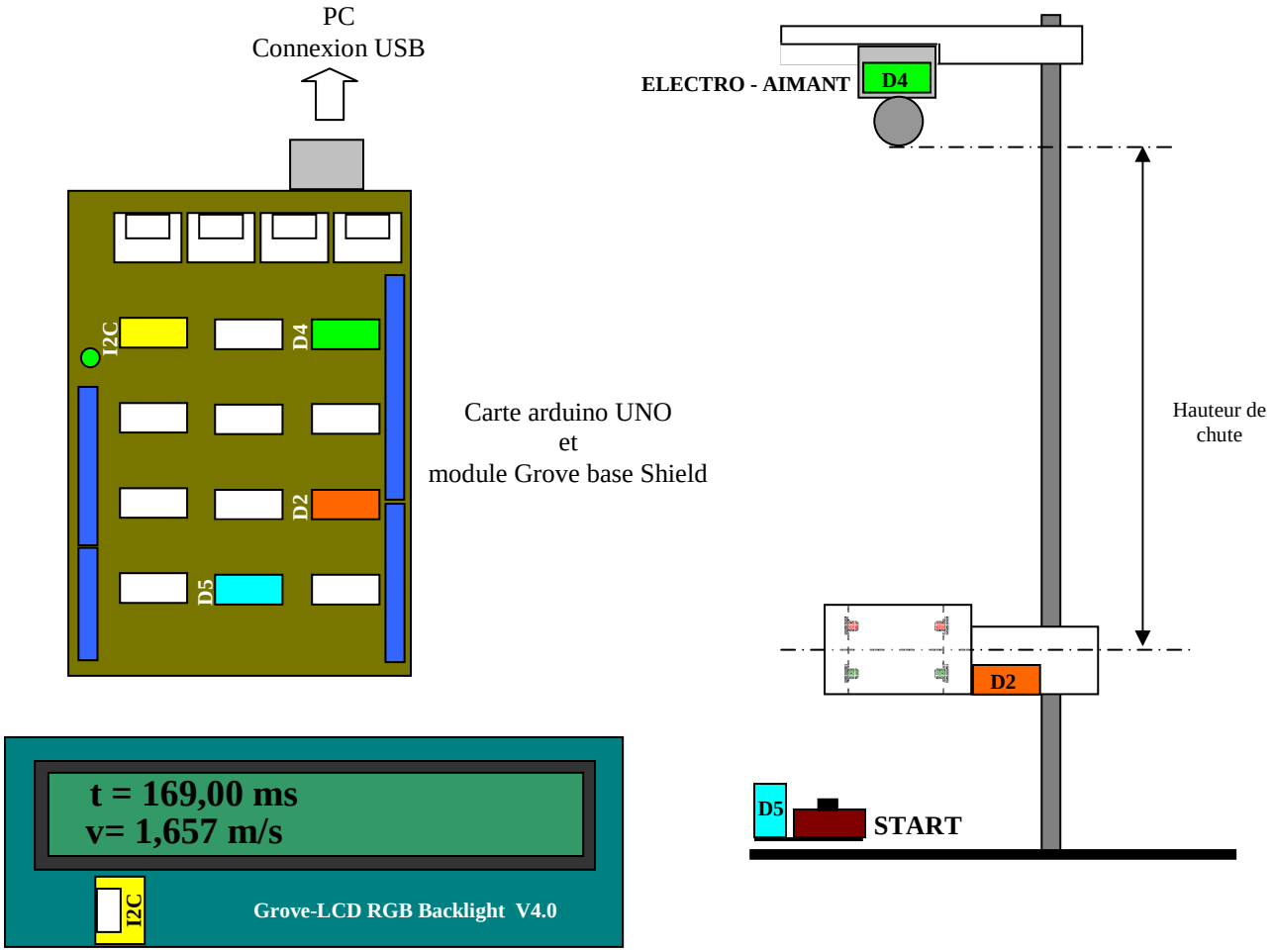


P.Langlois Mars 2019 / G.Eiffel
Pascal.langlois@ac-dijon.fr

Mesure de g : 9,89 $m.s^{-2}$

III. Mesure de la vitesse d'une bille en chute libre. Amélioration de la précision. Mesure de la durée de chute.

3.1 Montage :



3.2 Programme arduino (utilisation des interruptions):

```
#include <Wire.h>
#include <rgb_lcd.h>

rgb_lcd lcd;
const int colorR = 255;
const int colorG = 255;
const int colorB = 255;

const byte pinEA = 4;
const byte pinSTART = 5;
int VDH = 2;
```

```
int VDB = 3;
unsigned long T0;          // ajout d'un T0 correspondant au T start utilisé pour mesurer la durée de chute
unsigned long T1;
unsigned long T2;
float DT;
float V;
float t,t0,t1,t2;          // variables de temps converties en ms pour les calculs. Le T2 en µs donnant un
                           // nombre trop grand (débordement).

                                   // INTERRUPTIONS

                           // Causes d'interruptions liées à des niveaux ou des changements d'états
                           // des broches PD2 (INT0) ou PD3 (INT1)
ISR(INT0_vect)
{
   T1 = micros();          // Programme traité en cas d'interruption INT0
}

ISR(INT1_vect)
{
   T2 = micros();          // Programme traité en cas d'interruption INT1

}

void setup() {

lcd.begin (16,2);
lcd.setRGB (colorR, colorG, colorB);
Serial.begin(9600);

pinMode (pinEA,OUTPUT);
pinMode (pinSTART,INPUT);
pinMode (VDH,INPUT);
pinMode (VDB,INPUT);
T0 = 0;
T1 = 0;
T2 = 0;
                                   // INTERRUPTIONS
cli();                     // aucune interruption possible
EICRA &= 0xF0;             // EICRA  External Interrupt Control Register A . Reset du registre
EICRA = EICRA | 0x0A;      // écrire sur le registre A 1.0.1.0 (A en hexadécimale) afin de déclencher
                           // les IT sur front ↓
EIMSK |= 0x03;             // EIMSK External Interrupt  IT INT0 et IT INT1, activation des interruptions
sei();                     //  interruptions possibles


digitalWrite (pinEA,HIGH);
}

void loop() {

 if ((digitalRead(pinSTART)==HIGH)&(T0==0)){   // si le bouton start est ON et T0 = 0
                                               // alors
  delay(200);

    digitalWrite (pinEA,LOW);
```

```
   T0 = micros();
 }
 if ((T1 != 0) && (T2 != 0)){          // c.à.d si les acquisitions de T1 et T2 ont été réalisées
  DT = (float)(T2 - T1)/1000.00;       //
                                       //
  V = 0.0201*(1000.00/DT);             //          Calculs divers voir ci-dessus
  t1 = T1/1000;                        //
  t2 = T2/1000;                        //
  t0 = T0/1000;                        //

  t = (sqrt(((t1-t0)*(t1-t0)+(t2-t0)*(t2-t0))/2)-6);      // formule permettant de mesurer le temps passage de la bille
                                                          // au centre du capteur (-6 représentant le temps de réaction de
                                                           //l'EA) et d'en déduire la durée t de chute


 Serial.print("T0 = ");Serial.println(T0);          //
 Serial.print("T1 = ");Serial.println(T1);          //
 Serial.print("T2 = ");Serial.println(T2);          //
                                                    // Affichage divers (moniteur serie)
 Serial.print("DT = ");Serial.println(DT);          //
 Serial.print("V = ");Serial.println(V);            //
 Serial.print("t = ");Serial.println(t);            //

 Lcd.clear () ;

 lcd.setCursor (1,0);                               //
 lcd.print ("t = ");                                //
 lcd.print (t,2);                                   //
 lcd.print (" ms");                                 //
                                                    // Affichage LCD
 lcd.setCursor (0,1);                               //
 lcd.print ("v = ");                                //
 lcd.print (V,3);                                   //
 lcd.print (" m/s");                                //




 T1 = 0;
 T2 = 0;
 T0 = 0;

 delay(500);
 digitalWrite (pinEA,HIGH);
 }
}
```

3.3 Mesures:

z(m)	10 mesures de v (m/s)										**vitesse (m/s)****
0,05	0,987	0,987	0,987	0,987	0,987	0,987	0,987	0,988	0,988	0,988	**0,987**
0,1	1,4	1,401	1,399	1,4	1,4	1,4	1,399	1,399	1,4	1,4	**1,400**
0,15	1,711	1,711	1,711	1,711	1,713	1,711	1,713	1,711	1,711	1,71	**1,711**
0,2	1,987	1,985	1,988	1,987	1,987	1,987	1,987	1,987	1,987	1,987	**1,987**
0,25	2,222	2,22	2,219	2,222	2,22	2,222	2,22	2,221	2,222	2,221	**2,221**
0,3	2,432	2,434	2,431	2,431	2,432	2,433	2,433	2,432	2,433	2,43	**2,432**
0,35	2,623	2,619	2,704*	2,703*	2,632	2,63	2,622	2,623	2,623	2,622	**2,624**
0,4	2,804	2,883	2,804	2,805	2,802	2,802	2,804	2,807	2,801	2,811	**2,805**
0,45	2,97	2,957	3,059	2,965	2,974	2,972	2,963	2,975	2,972	2,965	**2,970**
0,5	3,117	3,137	3,102	3,131	3,131	3,125	3,121	3,121	3,123	3,121	**3,124**

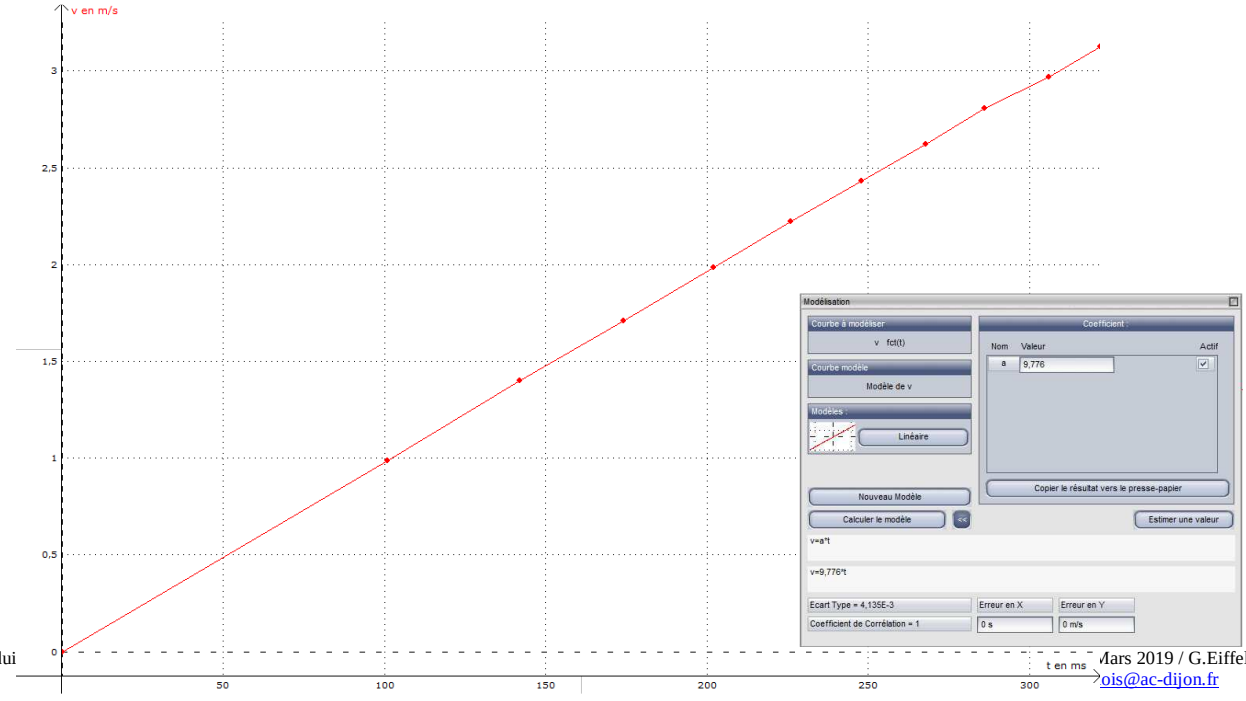
z(m)	10 mesures de t (s)										**durée de chute (s)****
0,05	0,102	0,101	0,1	0,102	0,103	0,102	0,1	0,1	0,101	0,101	**0,101**
0,1	0,144	0,141	0,142	0,143	0,142	0,142	0,142	0,148	0,141	0,142	**0,142**
0,15	0,174	0,174	0,174	0,173	0,173	0,177	0,174	0,173	0,174	0,177	**0,174**
0,2	0,205	0,203	0,201	0,201	0,203	0,201	0,201	0,202	0,201	0,202	**0,202**
0,25	0,226	0,225	0,226	0,226	0,225	0,225	0,226	0,226	0,226	0,225	**0,226**
0,3	0,247	0,249	0,249	0,247	0,248	0,249	0,249	0,249	0,25	0,246	**0,248**
0,35	0,268	0,276	0,267	0,267	0,267	0,268	0,268	0,267	0,267	0,269	**0,268**
0,4	0,287	0,286	0,288	0,286	0,285	0,285	0,285	0,286	0,286	0,286	**0,286**
0,45	0,305	0,32	0,307	0,312	0,303	0,304	0,308	0,303	0,306	0,304	**0,306**
0,5	0,326	0,319	0,335	0,32	0,32	0,322	0,324	0,321	0,321	0,322	**0,322**

* erreur de mesure

** moyenne sur huit mesures (la mesure la plus haute et la mesure la plus basse ne sont pas prisent en compte dans cette moyenne)

3.4 Courbe.

Vitesse en fonction du temps

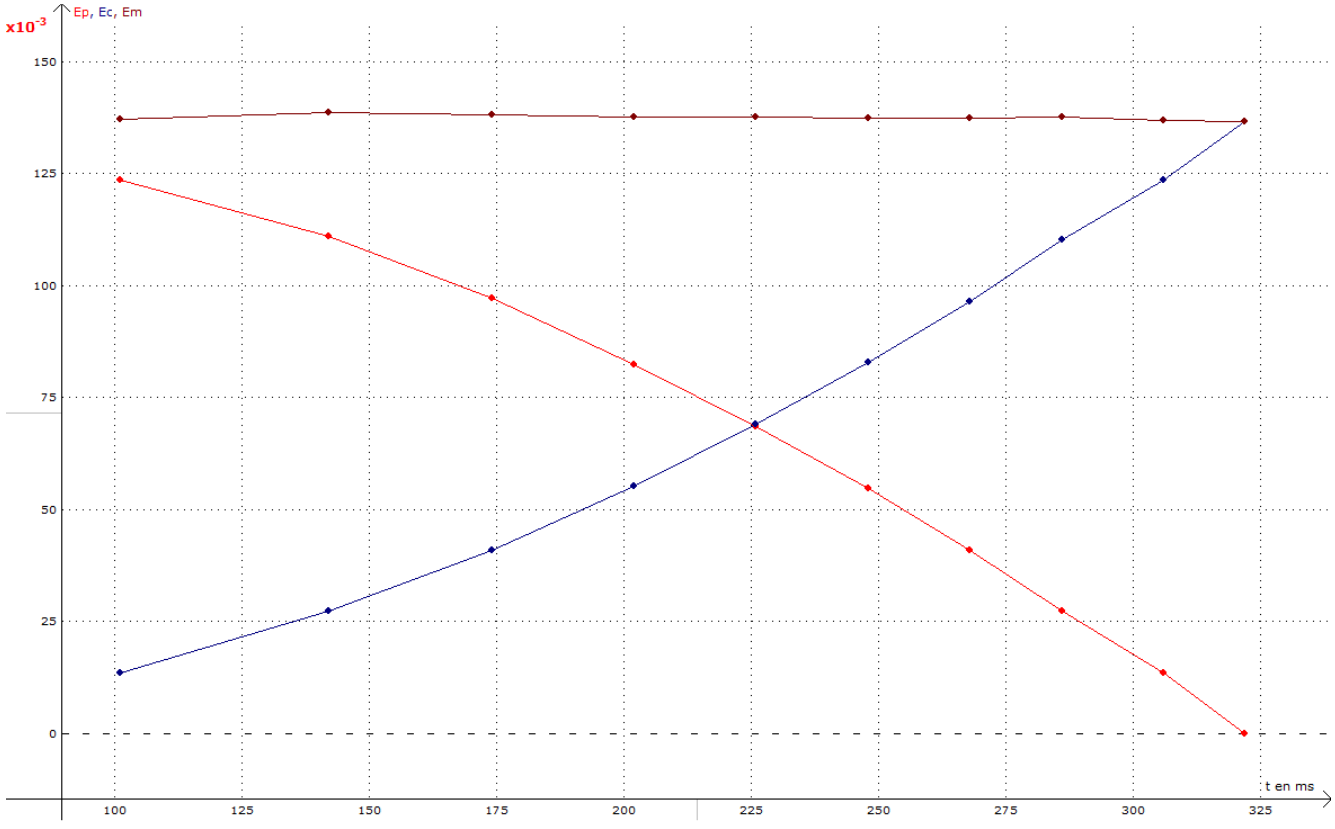


Mesure de g : 9,776 $m.s^{-2}$

3.5 Bilan énergétique :

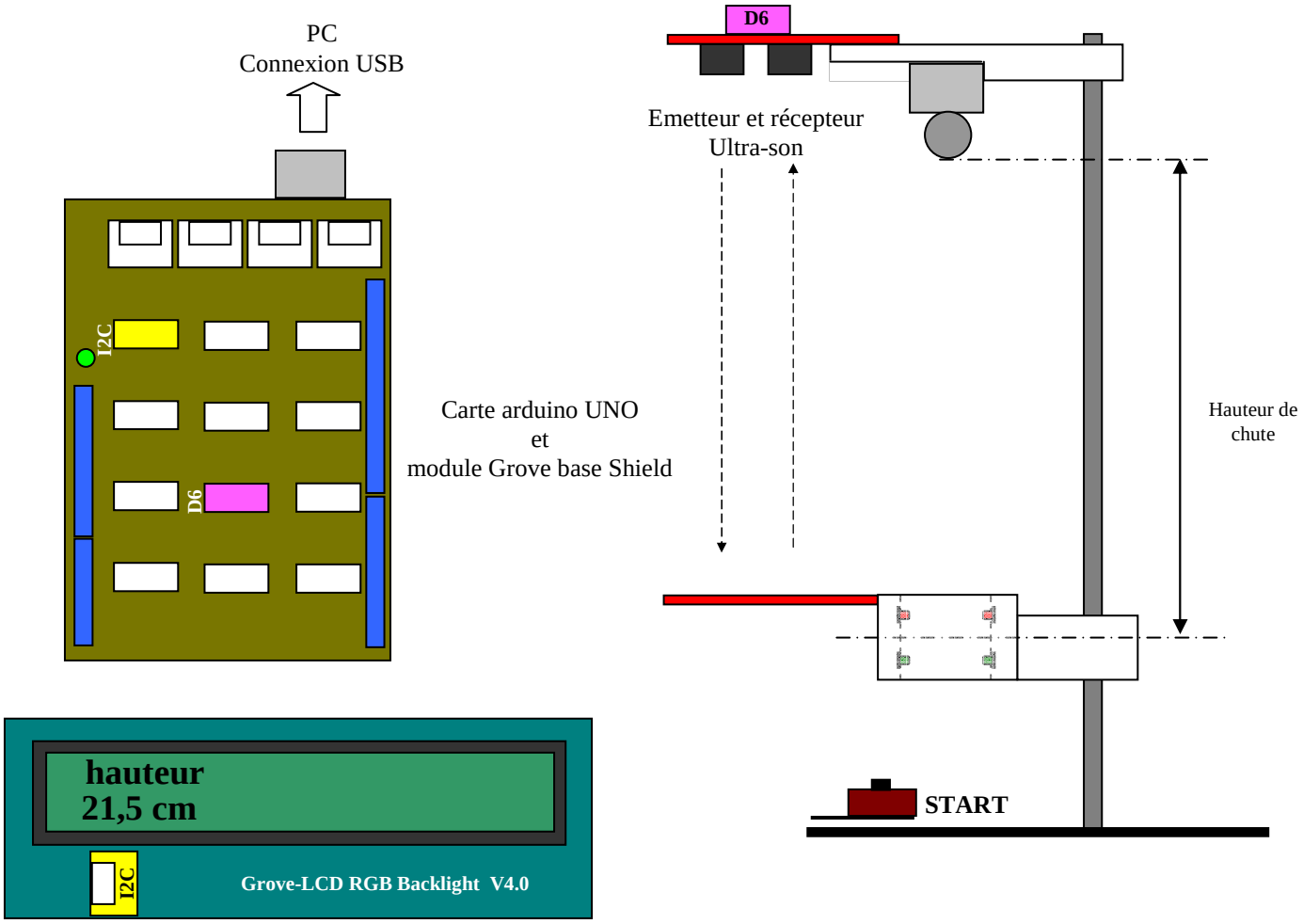
Programmation de la feuille de calcul de latis pro.

```
m=0.028                 //  poids de la bille (kg)
g=2*h/(t*t)             // calcul de g à partir des mesures
Ep=(-m)*g*(h-0.5)       // calcul de l'énergie potentielle. La référence est prise au niveau de
                        // de la dernière mesure (50cm)
Ec=0.5*m*V*V            // calcul de l'énergie cinétique
Em=Ep+Ec                // calcul de l'énergie potentielle
```



IV. Mesure de la hauteur de chute.

4.1 Montage :



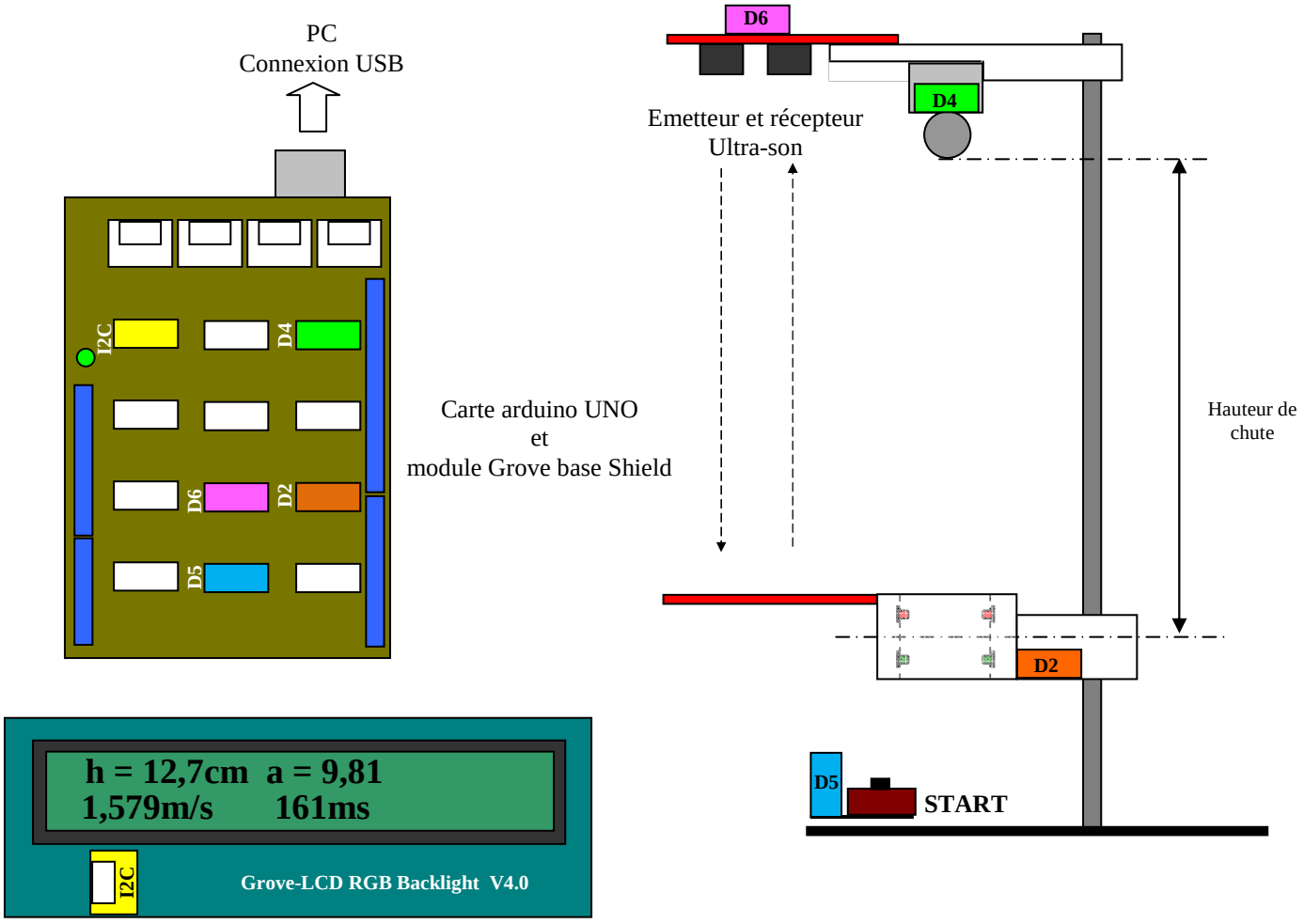
4.2 Programme arduino :

```
#include <Wire.h>
#include <rgb_lcd.h>

rgb_lcd lcd;
const int colorR = 255;
const int colorG = 255;
const int colorB = 255;

const byte pinTRIG = 7;      // déclaration des bornes du capteur US.
const byte pinECHO = 6;

const int nbrvaleur = 10;    // déclaration des variables utilisées dans un tableau de 10  mesures afin d'en
int temps[nbrvaleur];        // faire la moyenne
int readIndex = 0;
int total = 0;
```

```
float epbille = 1.9;
float duree = 0;
float hauteur = 0;

void setup() {

lcd.begin (16,2);
lcd.setRGB (colorR, colorG, colorB);

pinMode(pinTRIG,OUTPUT);
pinMode(pinECHO,INPUT);

for (int iniTab = 0; iniTab < nbrvaleur; iniTab++) // initialisation du tableau
  {
    temps[iniTab] = 0;
  }
  }


void loop() {

  total = total - temps[readIndex];

   digitalWrite(pinTRIG, LOW);
   delay(2);
   digitalWrite(pinTRIG, HIGH);                    // emission des US
   delay(10);
   digitalWrite(pinTRIG, LOW);
   temps[readIndex] = pulseIn(pinECHO,HIGH);  // mesure du temps de propagation
   total = total + temps[readIndex];               // Remplir le
   readIndex = readIndex + 1;                      //  tableau

   if (readIndex >= nbrvaleur)
   {readIndex = 0;}

   duree = total / nbrvaleur;                      // moyenne des mesures du tableau
   hauteur = (duree/58.4) - epbille + 1,1;         // calcul de la distance en cm "1,1 a ajuster en
                                                   // fonction de la position de l'émetteur récepteur  US"

   lcd.clear();

  lcd.setCursor (1,0);
  lcd.print ("hauteur");



  lcd.setCursor (0,1);
  lcd.print (hauteur,1);
  lcd.print ("cm");
}
```

V. Dispositif complet.
Mesure de la vitesse de chute.
Mesure du temps de chute.
Mesure de la hauteur de chute.
Mesure de l'accélération.

5.1 Montage :



Prévoir un afficheur plus grand pour pouvoir afficher la grandeur, l'unité et la valeur des quatre mesures.

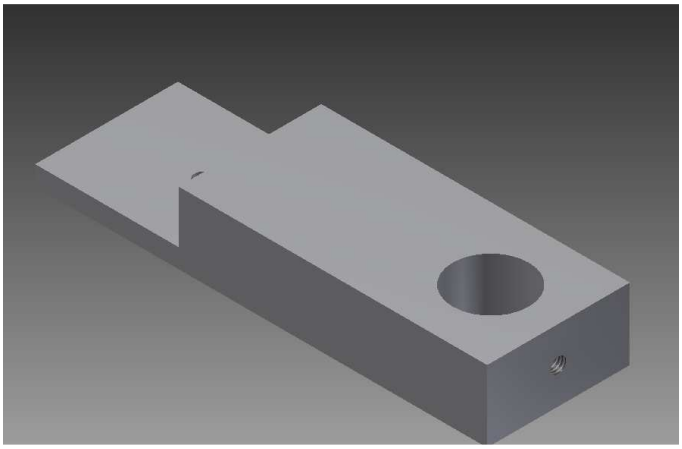
5.2 Programme complet :

```
#include <Wire.h>
#include <rgb_lcd.h>

rgb_lcd lcd;
const int colorR = 255;
const int colorG = 255;
const int colorB = 255;
```

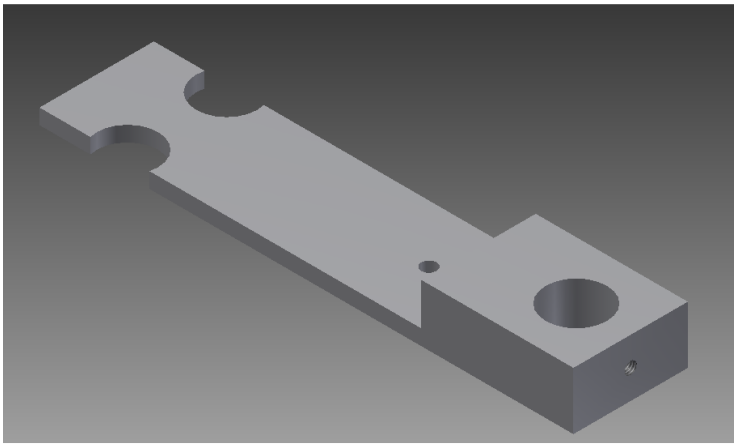
```
const byte pinEA = 4;
const byte pinSTART = 5;
const byte pinTRIG = 7;
const byte pinECHO = 6;


int VDH = 2;
int VDB = 3;
unsigned long T0;
unsigned long T1;
unsigned long T2;
float DT;
float V;
float t,t0,t1,t2;
const int nbrvaleur = 10;
int temps[nbrvaleur];
int readIndex = 0;
int total = 0;
float epbille = 1.9;
float duree;
float hauteur;
float acceleration;

ISR(INT0_vect)
{
   T1 = micros();
}

ISR(INT1_vect)
{
   T2 = micros();
}

void setup() {
  lcd.begin (16,2);
  lcd.setRGB (colorR, colorG, colorB);
  Serial.begin(9600);
  pinMode (pinEA,OUTPUT);
  pinMode (pinSTART,INPUT);
  pinMode (VDH,INPUT);
  pinMode (VDB,INPUT);
  pinMode(pinTRIG,OUTPUT);
  pinMode(pinECHO,INPUT);

  T0 = 0;
  T1 = 0;
  T2 = 0;

  cli();
  EICRA &= 0xF0;
  EICRA = EICRA | 0x0A;
  EIMSK |= 0x03;
  sei();

  for (int iniTab = 0; iniTab < nbrvaleur; iniTab++)
  {
    temps[iniTab] = 0;
  }

  digitalWrite (pinEA,HIGH);
}

void loop() {
```

```
 total = total - temps[readIndex];

  digitalWrite(pinTRIG, LOW);
  delay(2);
  digitalWrite(pinTRIG, HIGH);
  delay(10);
  digitalWrite(pinTRIG, LOW);
  temps[readIndex] = pulseIn(pinECHO,HIGH);
  total = total + temps[readIndex];
  readIndex = readIndex + 1;

  if (readIndex >= nbrvaleur)
  {readIndex = 0;}

  duree = total / nbrvaleur;
  hauteur=(duree/58.0) - epbille + 1.1;

  lcd.setCursor (1,0);
  lcd.print ("h=");lcd.print (hauteur,1);lcd.print ("cm");

  delay(10);

   if ((digitalRead(pinSTART)==HIGH)&(T0==0)){

  delay(200);
  digitalWrite (pinEA,LOW);

     T0 = micros();
}

 if ((T1 != 0) && (T2 != 0)){
   DT = (float)(T2 - T1)/1000.00;

   V = 0.0201*(1000.00/DT);
   t1 = T1/1000;
   t2 = T2/1000;
   t0 = T0/1000;

   t = (sqrt(((t1-t0)*(t1-t0)+(t2-t0)*(t2-t0))/2)-6);
   acceleration = V / (t / 1000);


 lcd.clear();

 lcd.setCursor (1,0);
 lcd.print ("h=");lcd.print (hauteur,1);lcd.print ("cm");lcd.print (" a=");lcd.print (acceleration,2);

 lcd.setCursor (0,1);
 lcd.print (V,3);
 lcd.print ("m/s ");

 lcd.print ( t,0);
 lcd.print ("ms");

 T1 = 0;
 T2 = 0;
 T0 = 0;
 delay(500);
 digitalWrite (pinEA,HIGH);
  }
}
```

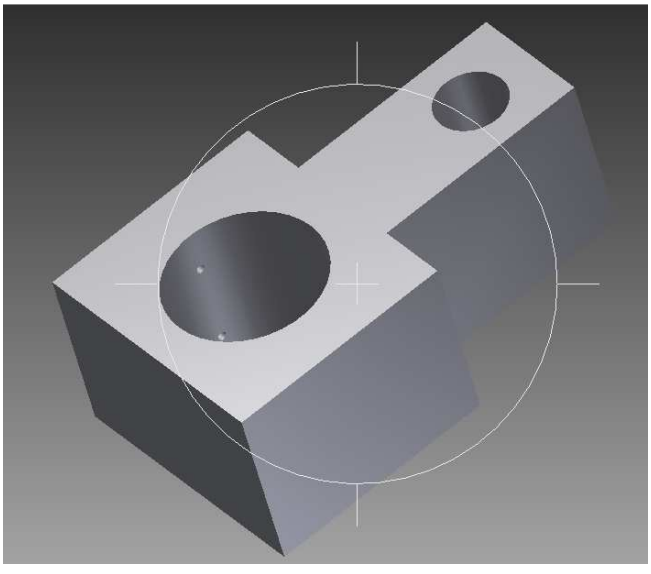
Annexe :
Impression 3d



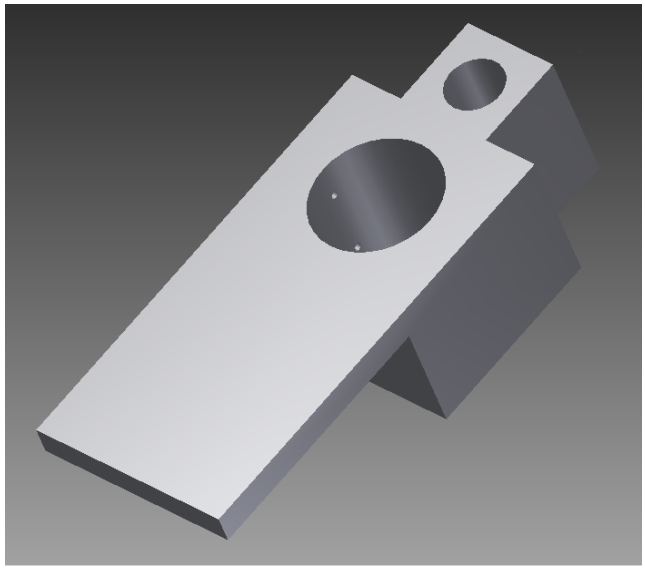
Support électro aimant



Support d'électro aimant avec émetteur récepteur US



Capteur infrarouge



Capteur infrarouge avec réflecteur US

P.Langlois Mars 2019 / G.Eiffel
Pascal.langlois@ac-dijon.fr

DEVIS

1	Carte arduino UNO https://www.gotronic.fr/art-carte-arduino-uno-12420.htm	19,50 € ttc
1	Module grove Base Shield https://www.gotronic.fr/art-module-grove-base-shield-103030000-19068.htm	09,30 € ttc
1	Module de détection US https://www.gotronic.fr/art-module-de-detection-us-hc-sr04-20912.htm	03,90 € ttc
1	Electroaimant Grove https://www.gotronic.fr/art-electroaimant-grove-101020073-21548.htm	10,40 € ttc
2	Module bouton Grove https://www.gotronic.fr/art-module-bouton-grove-111020000-19010.htm	03,98 € ttc
1	Afficheur LCD I2C Grove https://www.gotronic.fr/art-afficheur-lcd-i2c-grove-104030001-21308.htm	14,95 € ttc
2	Diode infrarouge https://www.conrad.fr/p/diode-infrarouge-ir-940-nm-30-3-mm-sortie-radiale-tru-components-1577525	00,80 € ttc
2	Phototransistor https://www.conrad.fr/p/phototransistor-lite-on-ltr-4206-3-mm-rond-940-nm-20-1-pcs-1127702	01,12 € ttc