Projet sciences de l'ingénieur : Aérofrein communicant sur un planeur

JACQUEMIN Gauthier, LINARD Clément,
JEUDY Mathis, FOURNIER Romain

Le planeur : un sport

2 types de compétitions

Course

- Lecture des courants d’air
- Rapidité et précision
- Navigation et stratégie
- Gestion de l’énergie

Voltige

Un sport qui rassemble :

Compétitions mixte

Handiplaneur

Fédération Française de Vol en Planeur

→ **157 clubs, + 13000 pilotes**

Origine du problème

Un pilote de planeur dans son cockpit ne possède aucune information sur la position de ses aérofreins et doit perpétuellement lors d'une phase d'atterrissage vérifier leurs positions avec un contact visuel direct. Ce qui risque de déconcentrer et de gêner le pilote lorsqu'il atterrit.

Problematique :

Comment développer un système permettant la communication de la position de l’aérofrein sur un planeur dans le cockpit ?

Analyse de l’existant :

Aucun système n’existe, seul le contact visuel permet de vérifier la sortie des aérofreins.

Répartition des tâches

Elèves	Répartitions des tâches	Définition des tâches			
Mathis JEUDY	Choix, étude et conception d'un système de détection du déploiement de l'aérofrein	Étude, choix et mise en œuvre d'un capteur.	Modélisation et simulation d'un support pour le capteur. Simulation et test du capteur	Programmation du microcontrôleur pour la gestion du capteur.	Réalisation de la présentation. Entrainement à la présentation orale et mesure des écarts, réalisation du dossier
Gauthier JACQUEMIN	Choix, étude, simulation et conception de la maquette de l'aile	Choix, étude et conception de l'intégration du système d'aérofrein dans la maquette de l'aile	Modélisation et simulation de la maquette principale		
Clément LINARD	Choix, étude et conception d'un système d'affichage	Etude, choix et mise en œuvre du système d'affichage	Modélisation et simulation du support de l'écran. Simulation et test du microcontrôleur et du système d'affichage	Programmation du microcontrôleur pour la gestion de l'affichage.	
Romain FOURNIER	Choix, étude, simulation et conception de l'intégration du système d'aérofrein dans l'aile	Choix et étude de l'intégration du système d'aérofrein dans la maquette de l'aile	Modélisation et simulation de l'intégration du système d'aérofrein dans la maquette de l'aile		

Diagramme des cas d'utilisation :

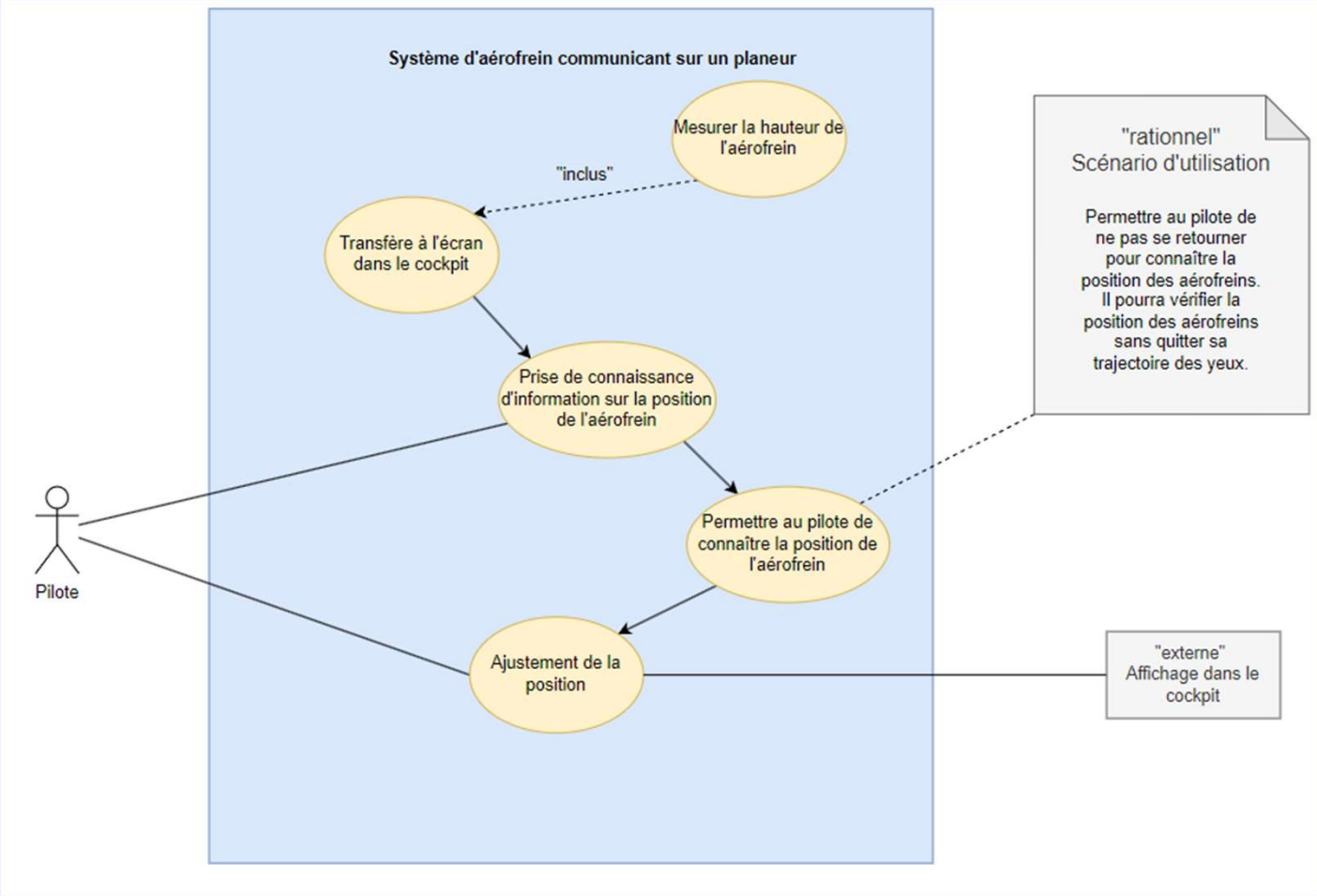
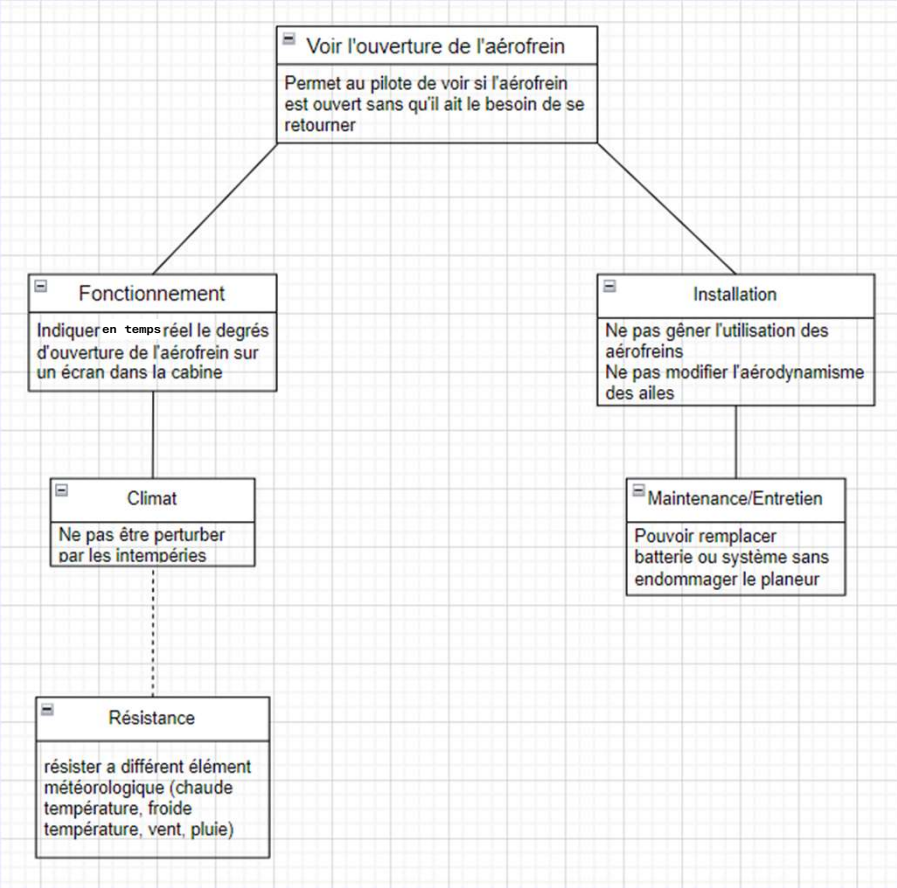


Diagramme des exigences:



Système de détection du déploiement de l'aérofrein :

Miroir / rétroviseur

- Permet un contact visuel directe

- Reflet du soleil dans les yeux du pilote

Capteur dans l'aérofrein

- Permet la mesure de la sortie des aérofreins

- Réglementation

Capteur sur la poigné

- Facilité d'installation

- Manque de sécurité

SOLUTION B

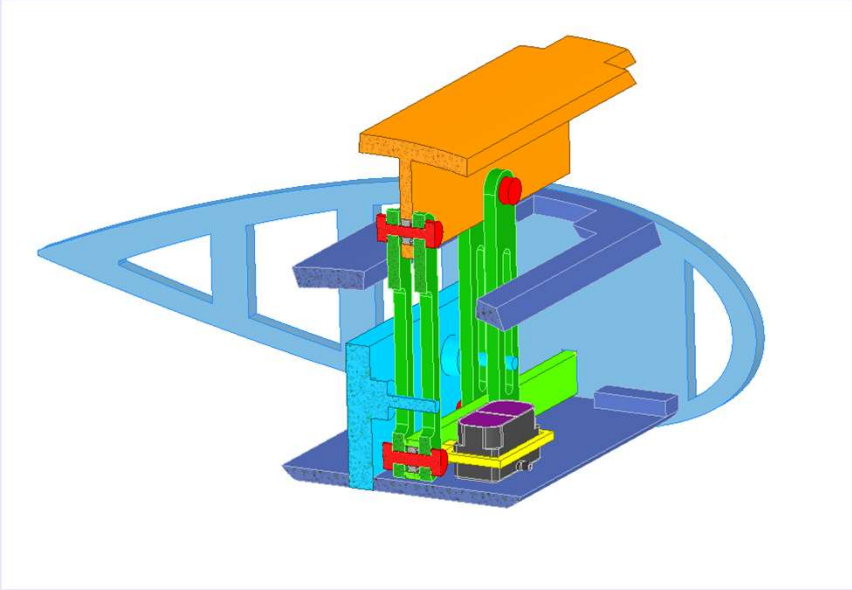
Étude et choix d'un capteur.

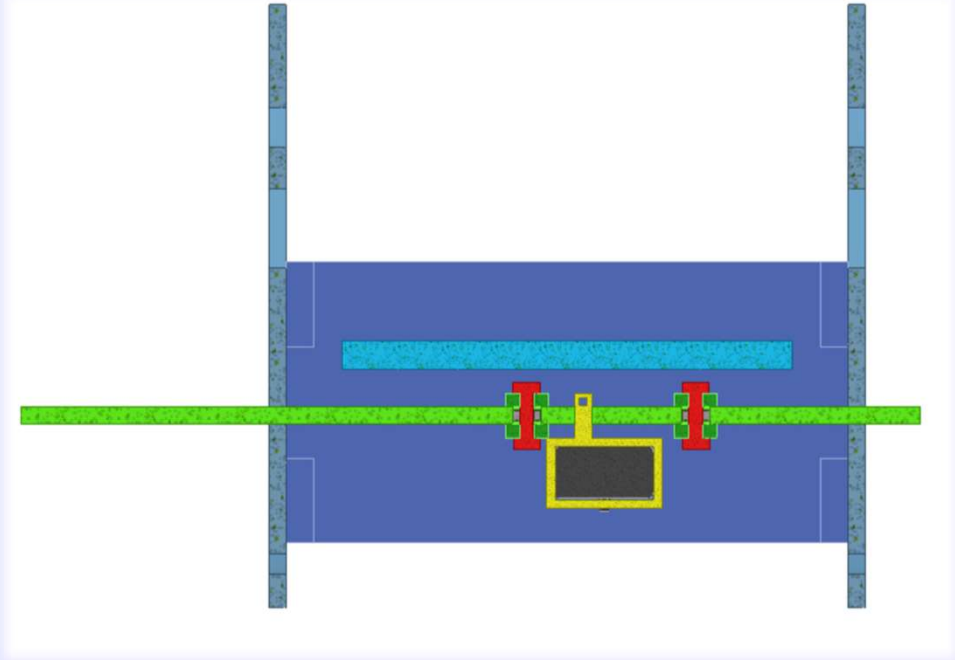
Différents capteurs :	Tension d'utilisation	Plage de mesure	Dimensions	Interface	Angle de mesure	Fréquence de mesure	Image
Capteur LIDAR	4,5 à 6 Vcc	De 30 cm à 1 200 cm	45x15x16 mm	Compatible Grove		100 Hz	
Capteur à ultrason	3,3 à 5 Vcc	De 2 cm à 400 cm	43x25x15 mm	Compatible Grove	15°	42 Hz	
Capteur télémètre infrarouge	3,3 à 5 Vcc	De 0 cm à 80 cm	40x20 mm	Compatible Grove		Infrarouge	
Capteur LIDAR Tf mini Plus	5 Vcc	De 2 cm à 1 200 cm	35x21x18,5 mm	Compatible Grove		1 à 1000 Hz	

Mise en œuvre du capteur.

- **Un système de fixation adapté à notre maquette**

- **Facilité de fixation**





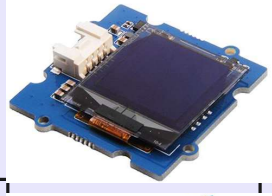
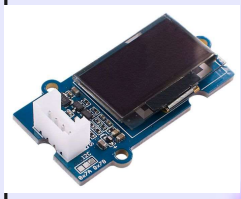
Système d’affichage :

Emplacement disponible

Permet au pilote de voir si l’aérofrein est ouvert sans qu’il ait le besoin de se retourner

Transmission de l’information de manière analogique pour pus de précision

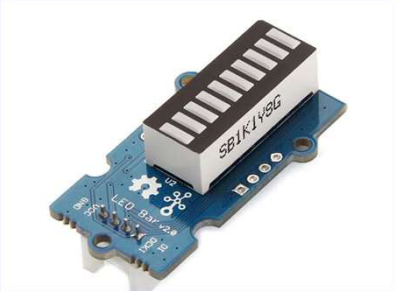
Etude et choix du système d'affichage :

Type	Tension d'utilisation	Affichage	Dimensions	Température de service	Poids	Interface	Image
Module OLED 1,12" 128 x 128 V3	3,3 ou 5 Vcc	128 x 128 pixels	40x40x12mm	-40°C à +70°C	15g	Compatible Grove	
Afficheur LCD 2x16 Grove	3,3 ou 5 Vcc	LCD 2x16	80x40x13 mm	0°C à +50°C	42,9g	Compatible Grove	
Grove - Écran OLED 0,96 pouces	3,3 ou 5 Vcc	128 x 64 pixels	42x24x9 mm	-20°C à +70°C	10g	Compatible Grove	
Module bargraphe Grove	3,3 ou 5 Vcc	Barre LED	40x20x20 mm	-20°C à +80°C	12g	Compatible Grove	

Etude, choix du système d'affichage :

Afficheur LCD 2x16 Grove

BARRE LED



Module OLED 1,12" 128 x 128 V3

Grove – Écran OLED 0,96 pouces

Système d’affichage

Codes pour la carte Arduino

LIDAR_Bar_graph

```
#include <SoftwareSerial.h>  //inclus la librairie du LIDAR
#include <Grove_LED_Bar.h> //inclus la barre LED

SoftwareSerial Serial1(2,3); //On définit les ports de branchement du capteur LIDAR

Grove_LED_Bar bar(7, 6, 0, LED_BAR_10); // On initialise le port de branchement de la barre LED
int distance (1,12000); //actual distance measurements of LiDAR
int check; //save check value
int i;
int uart[9]; //save data measured by LiDAR
const int HEADER=0x59; //frame header of data package

void setup() {
Serial.begin(115200); //set bit rate of serial port connecting Arduino with computer
Serial1.begin(115200); //vitesse initialisé à 115200 BAUD pour le LIDAR
}
void loop() {
if (Serial1.available()) { //check if serial port has data input
  if(Serial1.read() == HEADER) { //assess data package frame header 0x59
  uart[0]=HEADER;
  if (Serial1.read() == HEADER) { //assess data package frame header 0x59
    uart[1] = HEADER;
    for (i = 2; i < 9; i++) { //save data in array
      uart[i] = Serial1.read();
}
check = uart[0] + uart[1] + uart[2] + uart[3] + uart[4] + uart[5] + uart[6] + uart[7];
if (uart[8] == (check&0xff)){ //verify the received data as per protocol
distance = uart[2] + uart[3] * 256; //établir la relation correspondante pour la variable distance

  }
       }

 // Jauge de sortie aérofrein
  if(distance <= 2){
    bar.setLed(1, 0);// Si la distance est inférieur ou égale à 2cm alors éteindre la LED numéro 1
    }
  else{
    bar.setLed(1, 1);//Sinon allumer la LED numéro 1
    }
```

```
// Jauge de sortie aérofrein
 if(distance <= 2){
   bar.setLed(1, 0);// Si la distance est inférieur ou égale à 2cm alors éteindre la LED numéro 1
   }
 else{
   bar.setLed(1, 1);//Sinon allumer la LED numéro 1
   }

 if(distance <= 3){
   bar.setLed(2, 0);
   }
 else{
   bar.setLed(2, 1);
   }

 if(distance <= 4){
   bar.setLed(3, 0);
   }
 else{
   bar.setLed(3, 1);
   }

 if(distance <= 5){
   bar.setLed(4, 0);
   }
 else{
   bar.setLed(4, 1);
   }

 if(distance <= 6){
   bar.setLed(5, 0);
   }
 else{
   bar.setLed(5, 1);
   }

 if(distance <= 7){
```

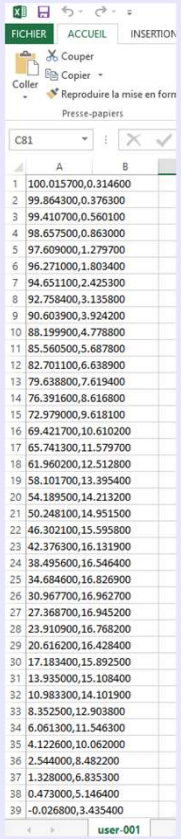
EMISSIOIN_R3_LATENCE

```
#include <SoftwareSerial.h>
#include <Grove_LED_Bar.h>
SoftwareSerial Serial1(3, 4); // Définit un port série logiciel avec les broches 3 (RX) et 4 (TX)
Grove_LED_Bar bar(7, 6, 0, LED_BAR_10); // Broche horloge, broche de données, orientation

int distance = 0; // Mesure de distance actuelle du LiDAR
int check; // Valeur de vérification
int uart[9]; // Données mesurées par le LiDAR
const int HEADER = 0x59; // En-tête de trame du paquet de données

void setup() {
  Serial.begin(115200); // Initialise la communication série avec le moniteur série à 9600 bauds
  Serial1.begin(115200); // Initialise la communication série avec le LiDAR à 9600 bauds
}

void loop() {
  if (Serial1.available()) { // Vérifie s'il y a des données disponibles sur Serial1
    if (Serial1.read() == HEADER) { // Vérifie le premier octet pour l'en-tête
      if (Serial1.read() == HEADER) { // Vérifie le deuxième octet pour l'en-tête
        uart[0] = HEADER;
        uart[1] = HEADER;
        for (int i = 2; i < 9; i++) { // Enregistre les données dans le tableau
          uart[i] = Serial1.read();
        }
        check = uart[0] + uart[1] + uart[2] + uart[3] + uart[4] + uart[5] + uart[6] + uart[7];
        if (uart[8] == (check & 0xFF)) { // Vérifie les données reçues selon le protocole
          distance = uart[2] + uart[3] * 256; // Calcule la valeur de distance
          Serial.print("Distance = ");
          Serial.println(distance); // Affiche la valeur de la distance mesurée par le LiDAR
          // Envoie la distance mesurée au récepteur via Serial
          Serial.print(distance);
          Serial.print('\n'); // Ajoute un retour à la ligne pour faciliter la lecture côté récepteur
        }
      }
// Contrôle de la barre LED en fonction de la distance
      if (distance <= 2) {
        bar.setLed(1, 0);
      } else {
        bar.setLed(1, 1);
      }

      if (distance <= 3) {
        bar.setLed(2, 0);
      } else {
        bar.setLed(2, 1);
      }

      if (distance <= 4) {
        bar.setLed(3, 0);
      } else {
        bar.setLed(3, 1);
      }

      if (distance <= 5) {
        bar.setLed(4, 0);
      } else {
        bar.setLed(4, 1);
      }

      if (distance <= 6) {
        bar.setLed(5, 0);
      } else {
        bar.setLed(5, 1);
      }

      if (distance <= 7) {
        bar.setLed(6, 0);
      } else {
        bar.setLed(6, 1);
      }

      if (distance <= 8) {
```

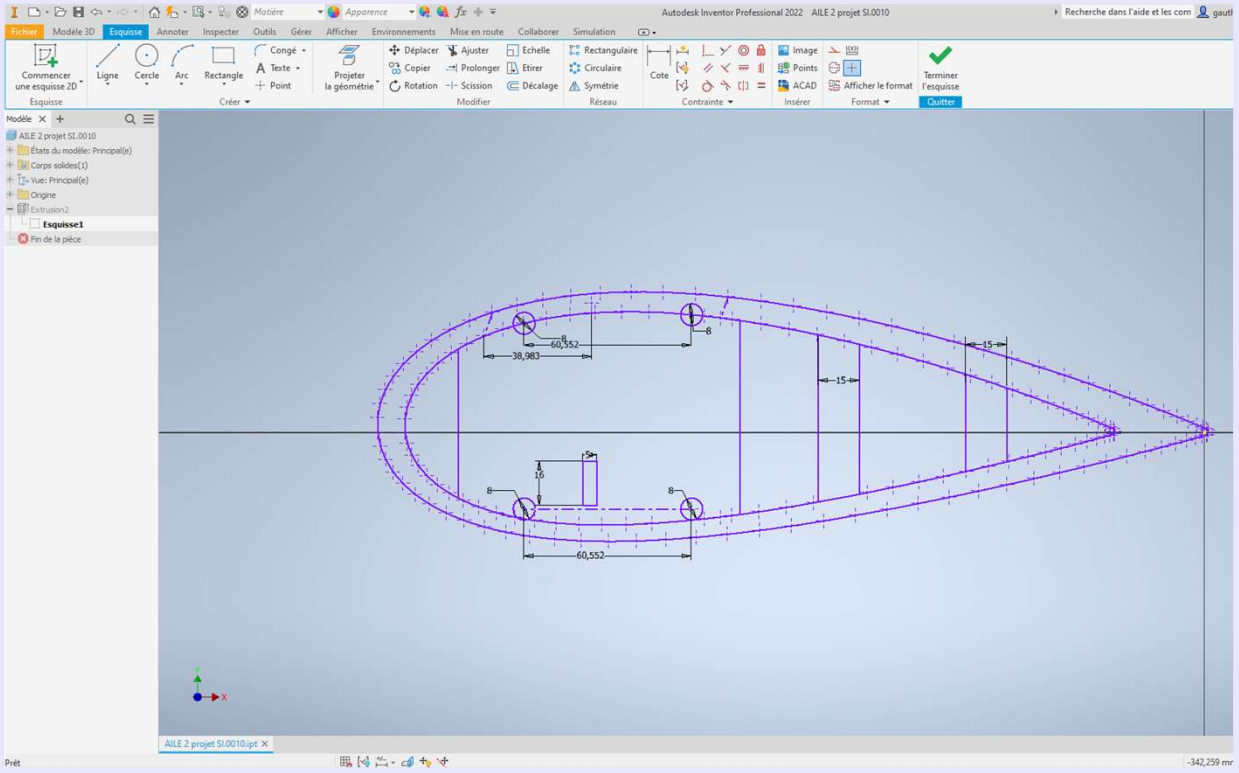
RECEPTION_R4_FINITO_LATENCE

```
#include <Arduino.h>
#include <U8g2lib.h>
#include <SPI.h>
#include <Wire.h>

U8G2_SH1107_SEEED_128X128_1_SW_I2C u8g2(U8G2_R0, /* clock=*/ SCL, /* data=*/ SDA, /* reset=*/ U8X8_PIN_NONE);

void setup() {
  Serial.begin(115200);  // Initialise la communication série à 9600 bauds
  Serial1.begin(115200); // Initialise la communication série avec l'émetteur à 9600 bauds
  u8g2.begin();
}

void loop() {
  if (Serial1.available()) { // Vérifie s'il y a des données disponibles sur le port série
    int distance = Serial1.parseInt(); // Lit la distance envoyée par l'émetteur


     //contrôle de l'écran
     u8g2.firstPage();
         do {
          u8g2.setFont(u8g2_font_inb42_mn);
           u8g2.setCursor(0,90);
           u8g2.print(distance);
         } while ( u8g2.nextPage() );
   }
  }
```

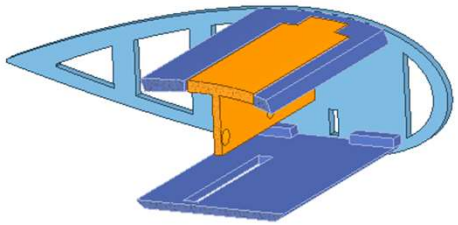
Choix et étude de la maquette de l'aile :

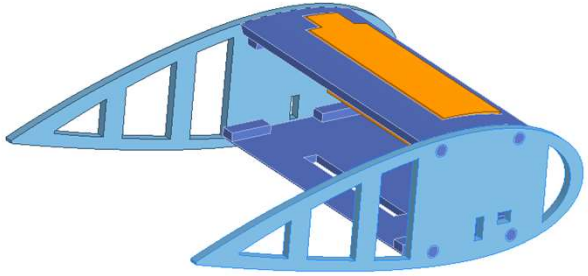


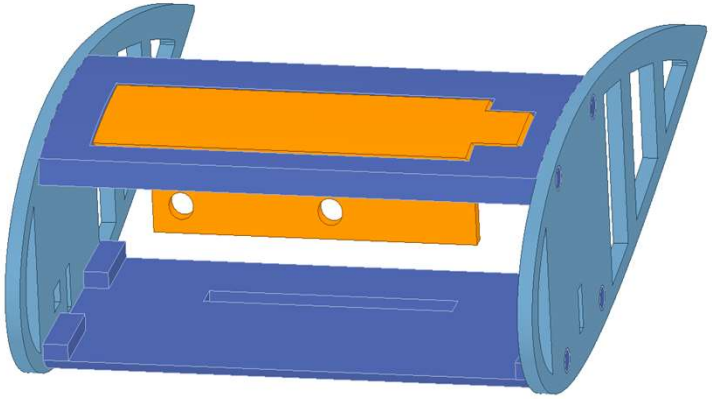
	A	B
1	100.015700,0.314600	
2	99.864300,0.376300	
3	99.410700,0.560100	
4	98.657500,0.863000	
5	97.609000,1.279700	
6	96.271000,1.803400	
7	94.651100,2.425300	
8	92.758400,3.135800	
9	90.603900,3.924200	
10	88.199900,4.778800	
11	85.560500,5.687800	
12	82.701100,6.638900	
13	79.638800,7.619400	
14	76.391600,8.616800	
15	72.979000,9.618100	
16	69.421700,10.610200	
17	65.741300,11.579700	
18	61.960200,12.512800	
19	58.101700,13.395400	
20	54.189500,14.213200	
21	50.248100,14.951500	
22	46.302100,15.595800	
23	42.376300,16.131900	
24	38.495600,16.546400	
25	34.684600,16.826900	
26	30.967700,16.962700	
27	27.368700,16.945200	
28	23.910900,16.768200	
29	20.616200,16.428400	
30	17.183400,15.892500	
31	13.935000,15.108400	
32	10.983300,14.101900	
33	8.352500,12.903800	
34	6.061300,11.546300	
35	4.122600,10.062000	
36	2.544000,8.482200	
37	1.328000,6.835300	
38	0.473000,5.146400	
39	-0.026800,3.435400	



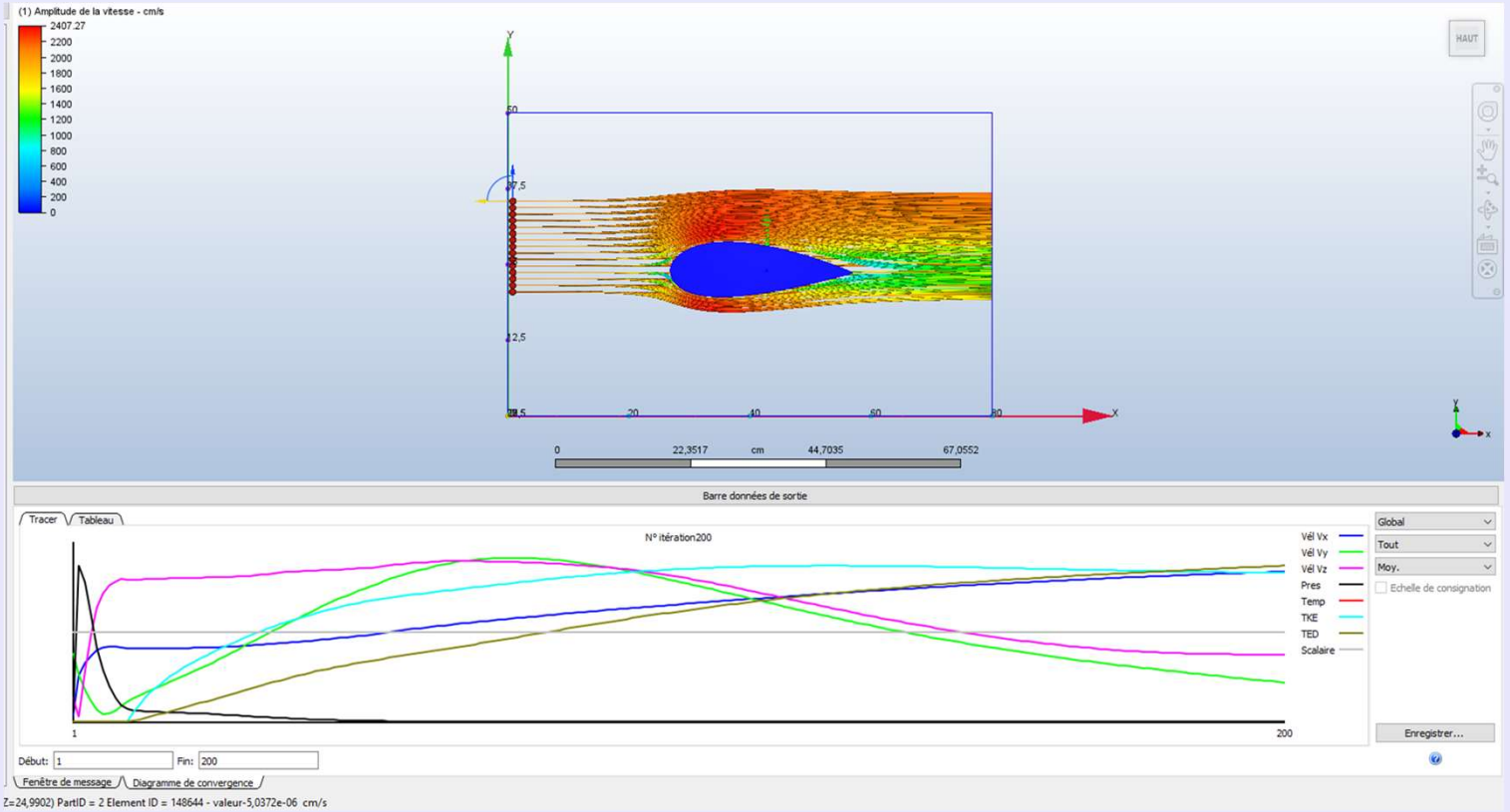
Conception de la maquette de l'aile :



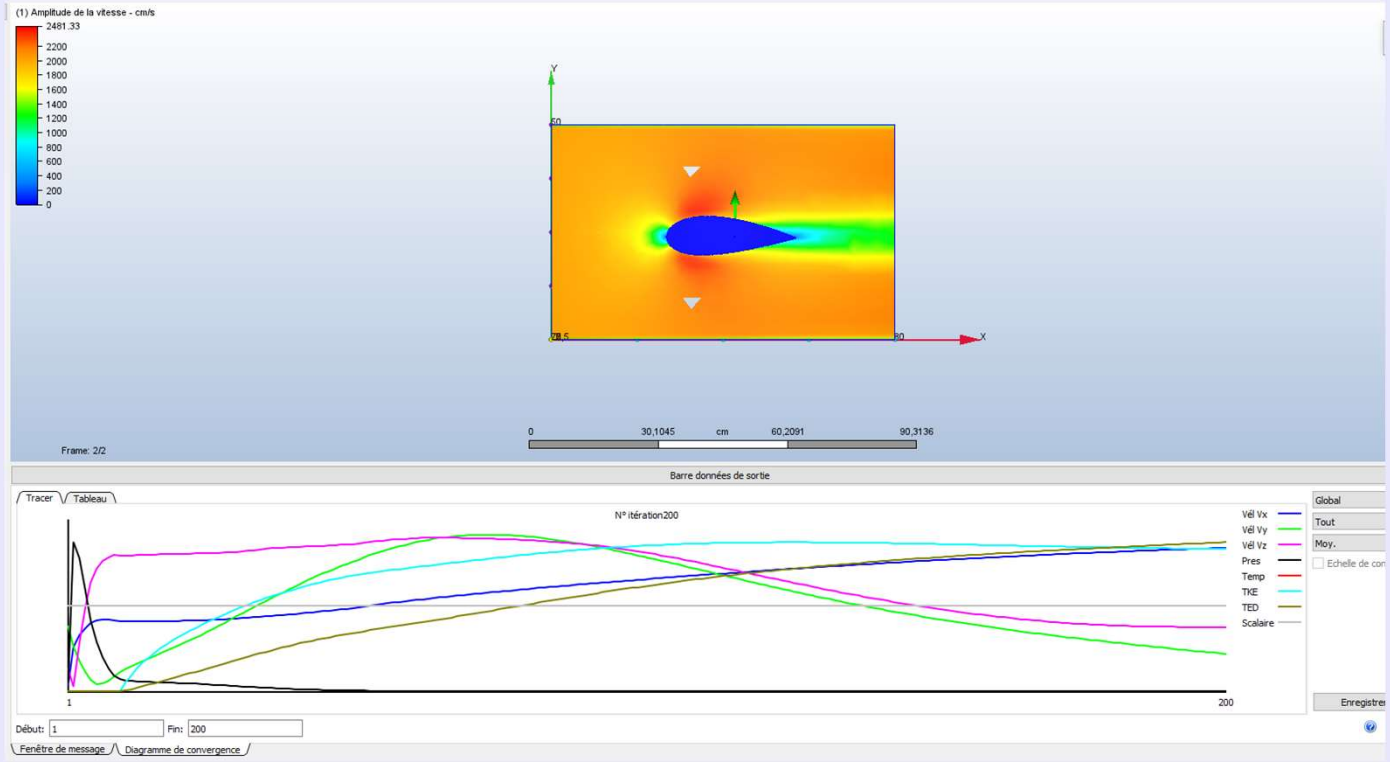




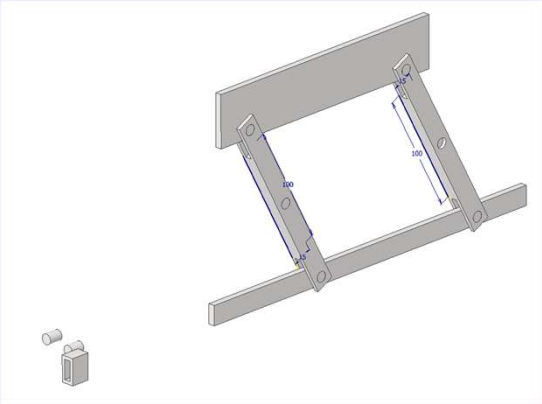
Simulation du profil de la maquette de l'aile

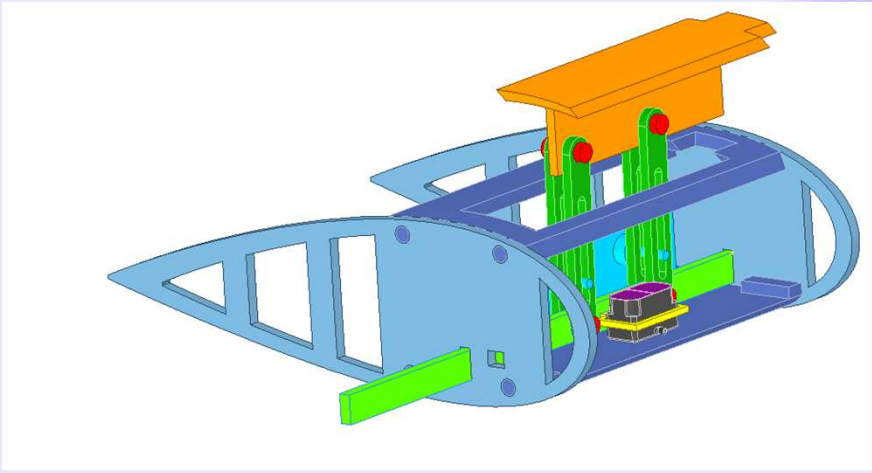


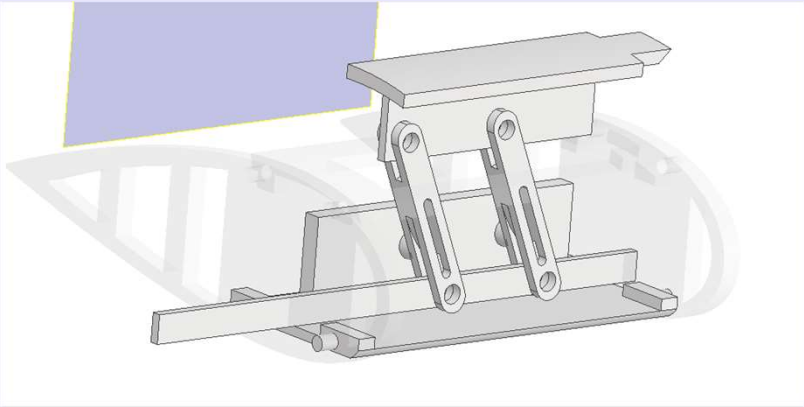
Simulation du profil de la maquette de l'aile



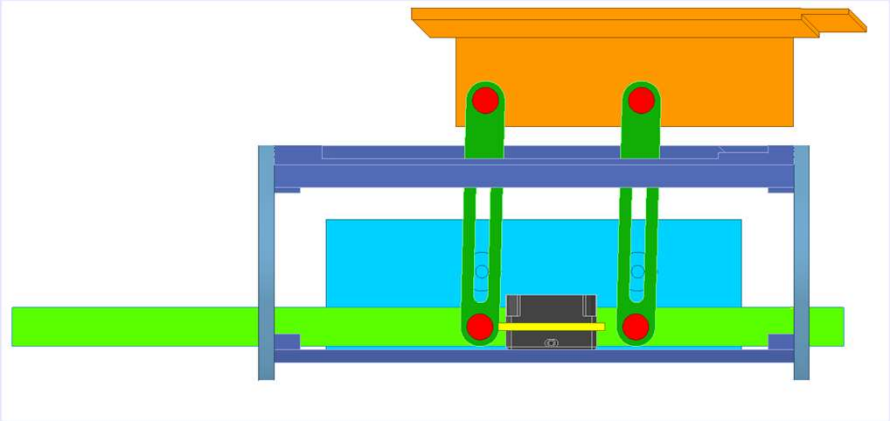
Choix, étude de l'intégration du système d'aérofrein dans l'aile :

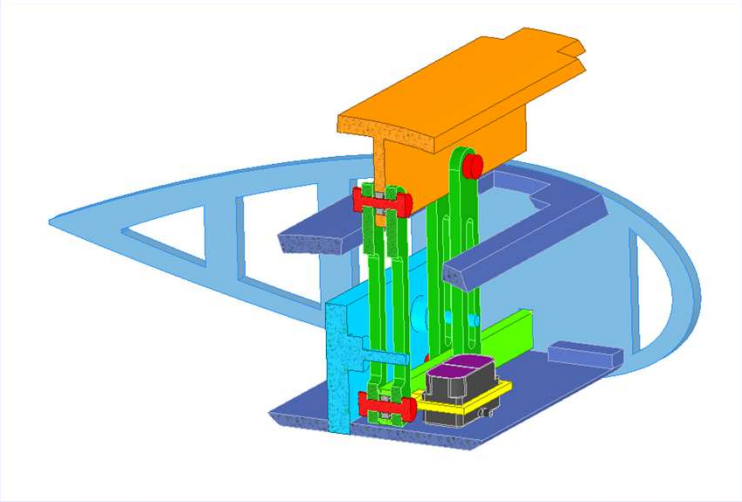


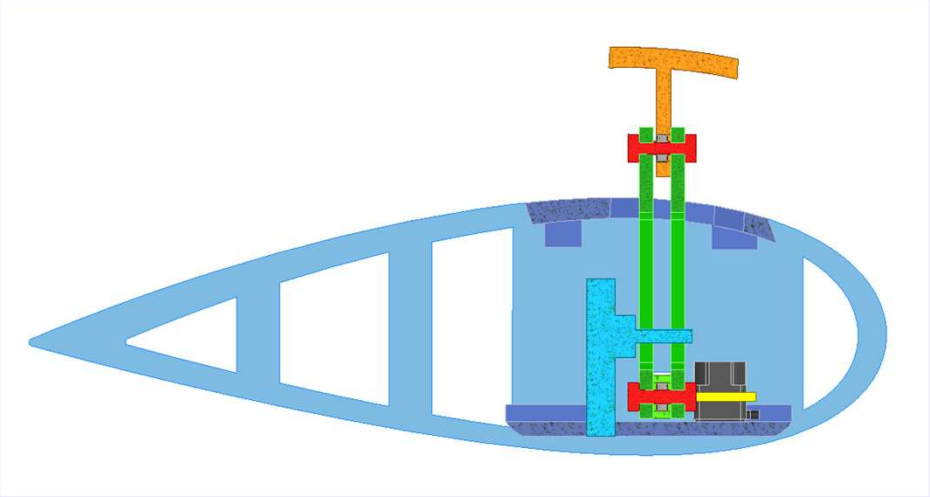




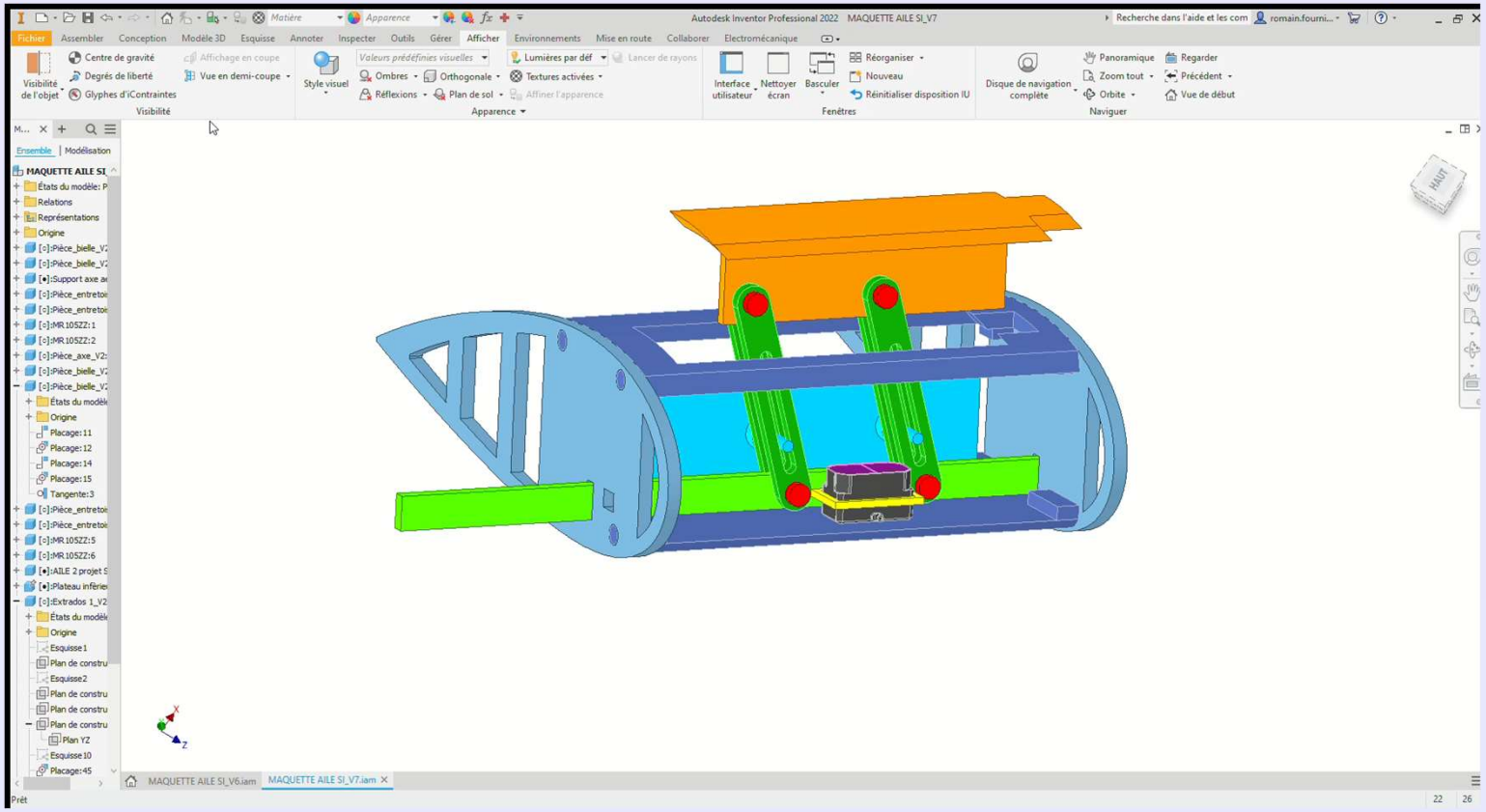
Simulation du système d’aérofrein dans l’aile :



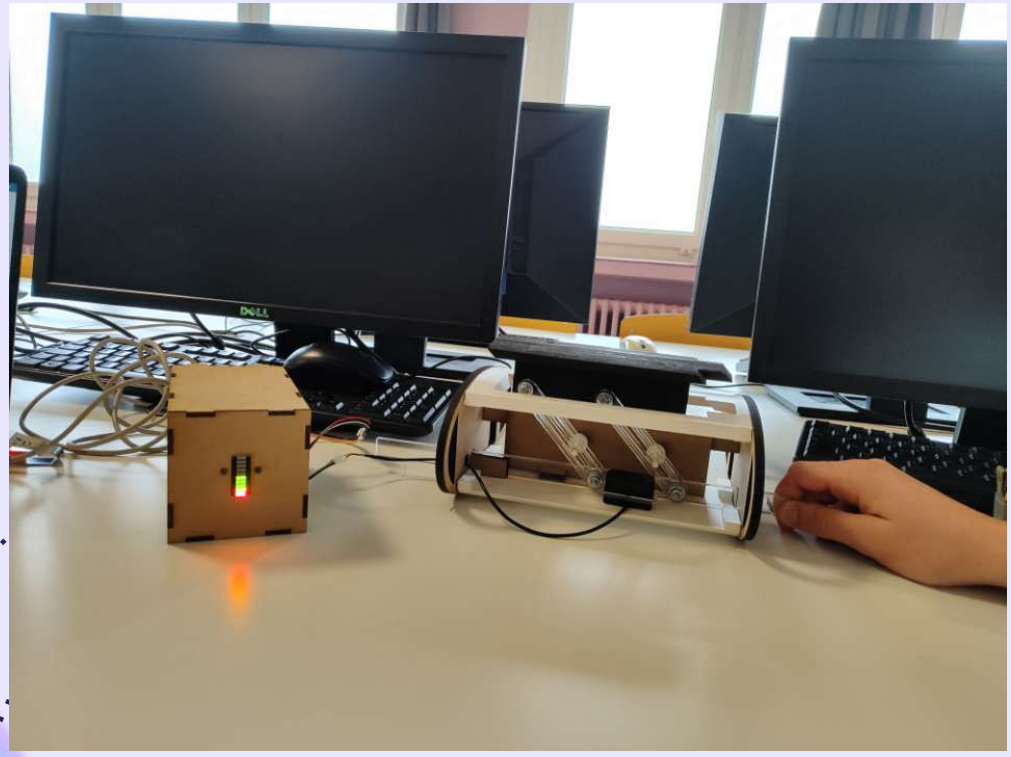




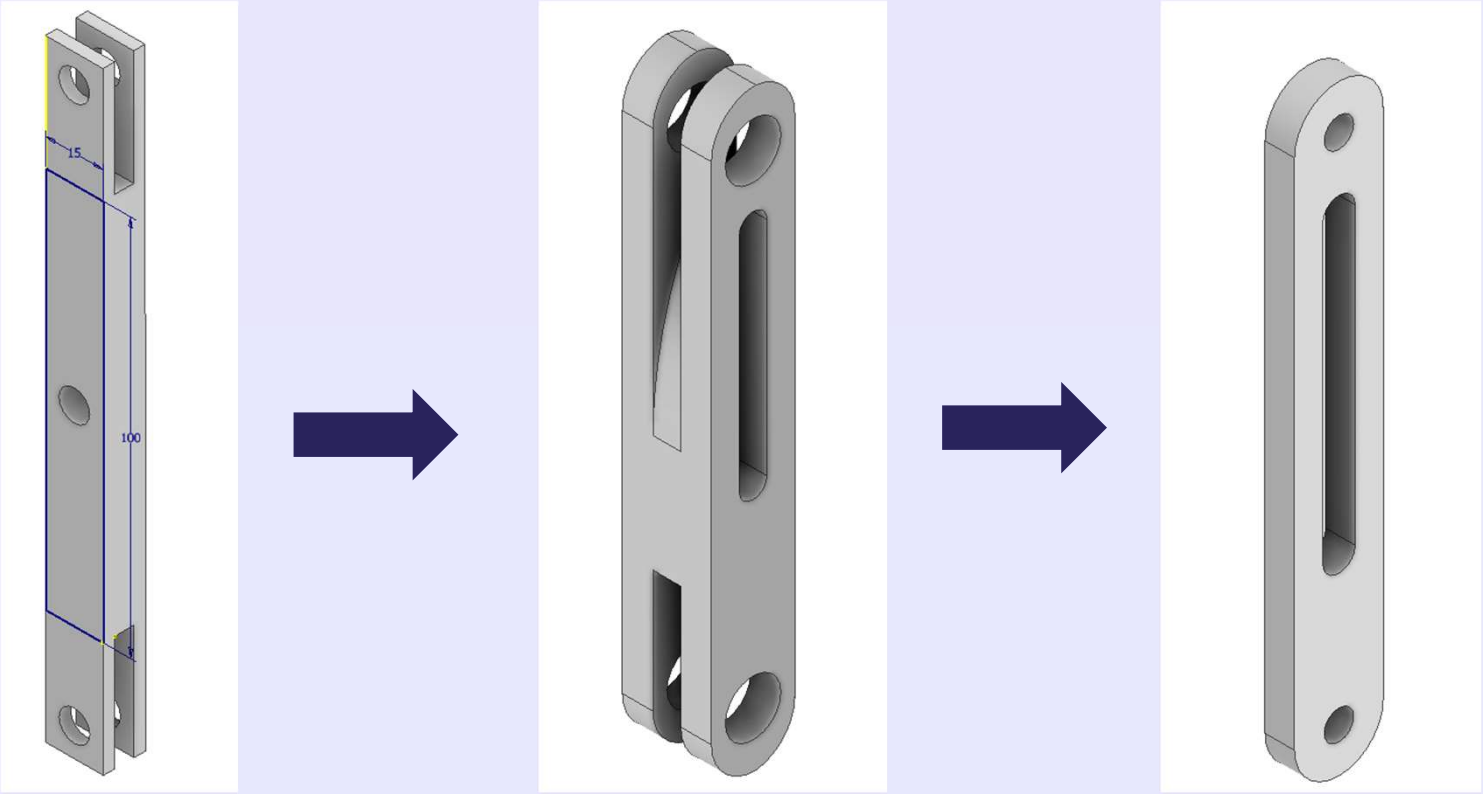
Simulation du système d’aérofrein dans l’aile :



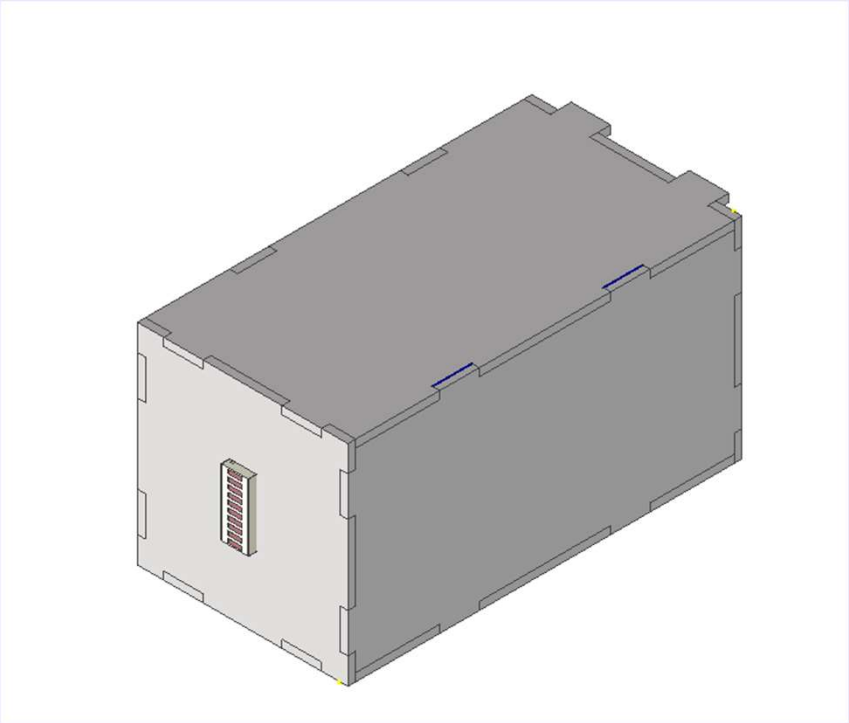
Maquette aérofrein :

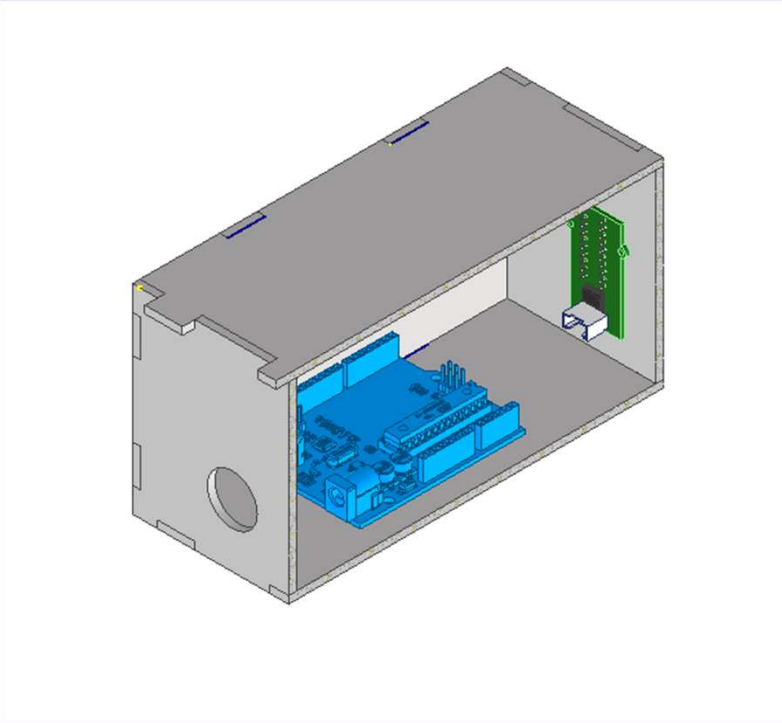


Conception du système d’aérofrein dans l’aile :

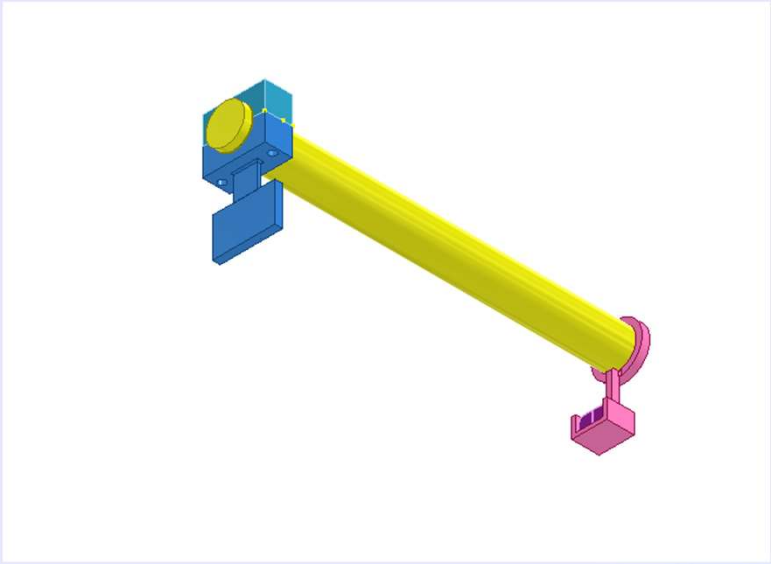


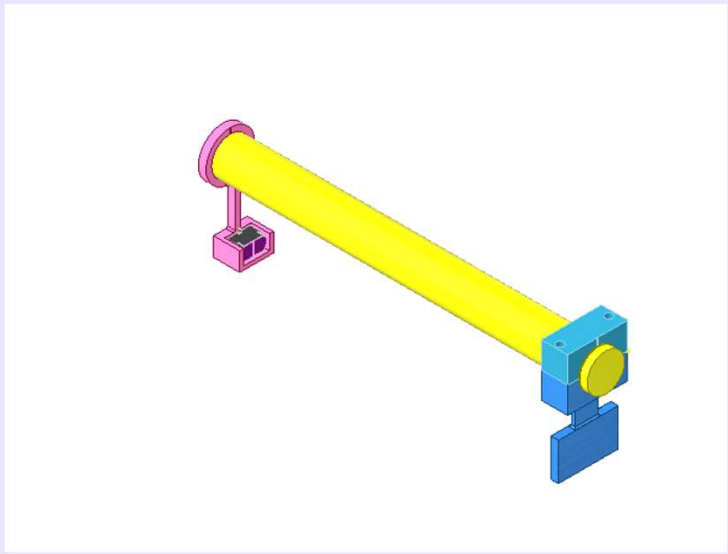
Conception du boitier d’affichage :





Solution B :





Bilan :

- Le pilote de planeur à l'information de la position des aérofreins de manière précise sans avoir le besoin de se retourner

- Ce système permettrait de faciliter les phases d'atterrissage.

- Augmente la sécurité avec plus d'informations à bord du cockpit

- Bénéfique lors de la pratique de la voltige en planeur