Caméra de Bord LoRa : Télémétrie d'Images pour la Fusée Projet thématique

Auteurs : EL MEADY FATIMAZAHRA
EL KABBAJ ANASS
Encadrants : GHIOTTO Anthony
FERRE Guillaume
Association partenaire : EIRSPACE
Concours : C'Space
Date de réalisation : 06/05/2025
2024/2025

Contents

1 Introduction

Le projet **FuseXX** s'inscrit dans le cadre de l'initiative **C'Space** organisée par Planètes des Sciences à Tarbes, dont l'objectif est de promouvoir l'ingénierie aérospatiale et l'exploration spatiale auprès des étudiants. Notre équipe, composée de huit étudiants de l'**ENSEIRB-MATMECA** répartis en quatre binômes, conçoit, assemble et lance une fusée expérimentale e. Chaque binôme couvre un domaine clé afin d'assurer le succès de la mission et de bénéficier d'une immersion technique complète.

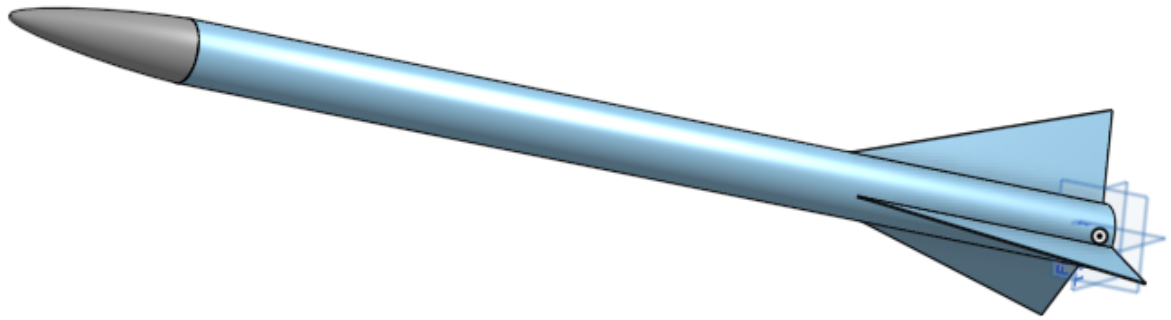
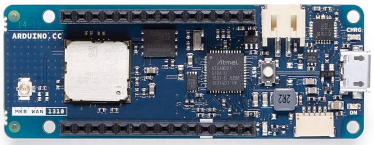


Figure 1: Fusexx Main CAD

Notre binôme est spécifiquement chargé de la transmission d'images à longue portée. Plutôt qu'un véritable flux vidéo, nous capturons des clichés à intervalles définis grâce à un module **ESP-CAM**, puis les transmettons au sol via la technologie **LoRa**. Cette approche permet de compenser la bande passante très limitée de **LoRa** tout en offrant une série d'instantanés exploitables pour reconstituer, en post-traitement, le déroulé visuel du vol. Initialement, nous avons utilisé un module **SX1276**, mais nous avons rencontré des difficultés de stabilité et d'intégration ; c'est pourquoi nous avons basculé vers la carte Arduino **MKR WAN**, qui s'est révélée plus fiable et mieux adaptée à nos besoins. Nous aborderons les raisons de ce choix plus en détail dans la suite du rapport.

Notre objectif est de démontrer qu'il est possible d'obtenir des retours visuels pertinents, depuis le décollage jusqu'à l'apogée, sans recours à un réseau Wi-Fi ou cellulaire, en s'appuyant uniquement sur la portée étendue et la faible consommation de **LoRa**.

(a) Module ESP32-CAM AI-Thinker



(b) MKR WAN 1310

(c) SX1276

Figure 2: Les composants clés de notre chaîne vidéo longue portée

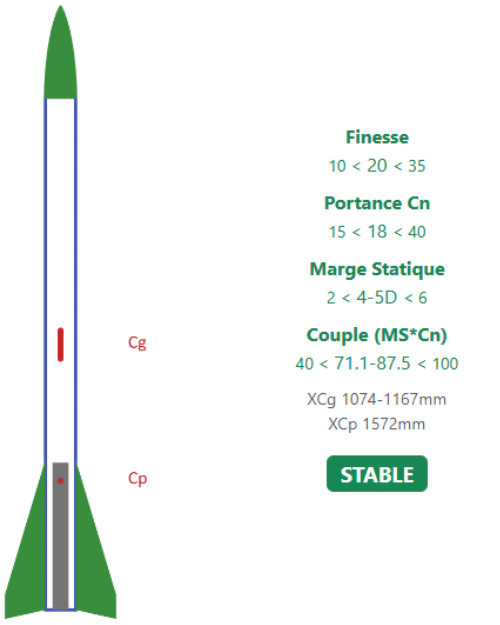
- OVERVIEW

  La structure de la fusée se compose donc d'un corps principal de 2 m de long pour un diamètre de 10 cm, aux extrémités équilibrées par un empennage métallique triangulaire de surface modulable. Le propulseur à propergol solide, délivrant près de 2 kN de poussée pendant 5 s, est positionné en fond de fuselage, tandis que la coiffe contient les différentes expériences scientifiques.

  Chaque binôme a pris en charge un sous-système critique :

  – Télémetrie capteurs analogiques
  – Traitement au sol
  – Conception antennes
  – Transmission/Reception d'Images (notre sujet)

  La simulation de FuseXX montre une montée rapide jusqu'à 1678 m en 17,7 s, suivie d'une phase balistique et d'une descente sous parachute sur 161,3 s au total, pour une portée d'environ 1018 m. Les courbes de vitesse et d'altitude confirment la performance du moteur, tandis que les paramètres clés valident le pic d'accélération de 107 m/s² et la vitesse de descente maîtrisée. L'analyse longitudinale révèle un centre de gravité toujours en avant du centre de pression, avec une marge statique de l'ordre de 4–5 D, assurant un vol stable et contrôlé.



(a) Centre de gravité vs. pression et marge statique.

(b) Paramètres clés du vol (vitesse, apogée, durée).

Figure 3: Synthèse de la trajectoire et de la stabilité de FuseXX.

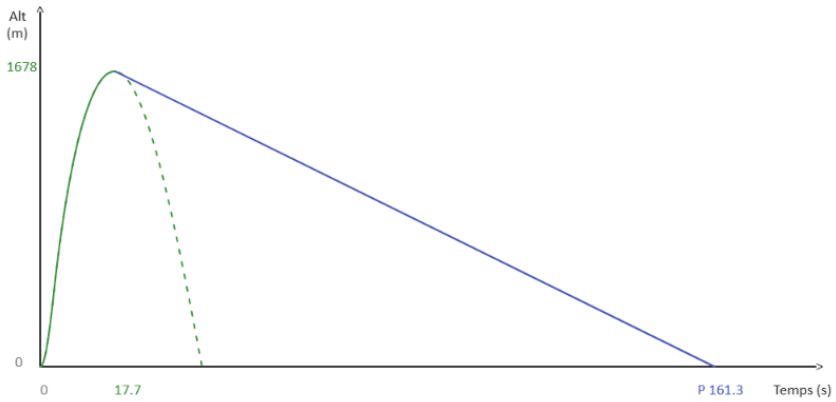


Figure 4: Altitude vs. temps.

Après cette présentation globale de **FuseXX** et du rôle de chaque binôme, ce rapport décrira en détail notre méthodologie : d'abord la capture et le chiffrement des images, puis leur fragmentation et transmission, et enfin leur réception et recomposition.

2 Solution intégrée de capture, transmission et réception d'images via SX1276

2.1 Chaîne de transmission d'instantanés LoRa

Dans un premier temps, nous avons mis en place un prototype associant un module **ESP32-CAM** et un transceiver **LoRa SX1276** pour tester la faisabilité de la transmission d'images en environnement contraint. L'objectif était de capturer un cliché **JPEG** à bord puis de le transmettre au sol malgré la bande passante réduite offerte par **LoRa**.

Le matériel se compose ainsi du module **ESP32-CAM** pilotant la caméra en résolution CIF, qualité JPEG réglée à 40, et du SX1276 connecté en SPI (broches SCLK, MISO, MOSI et SS). Afin de maximiser la consommation énergétique et les ressources disponibles, nous avons désactivé le Wi-Fi et le Bluetooth sur l'ESP32.

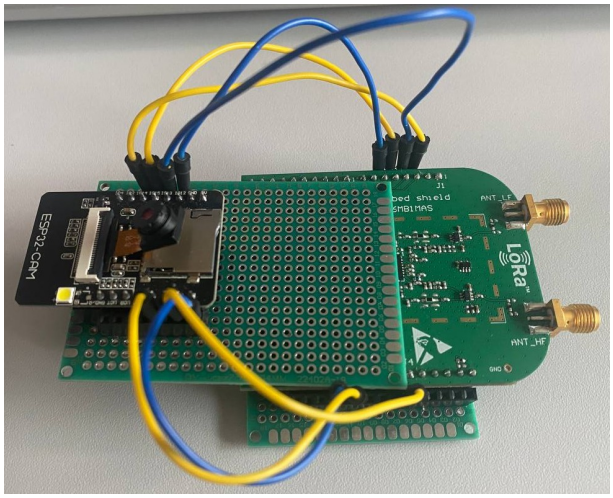


Figure 5: Altitude vs. temps.

2.1.1 Matériel

- **Module ESP32-CAM**
- **Transceiver LoRa SX1276**
- **Alimentation** 3.3 V via micro-USB de l'ESP32
- **Câbles Dupont** et breadboard

2.1.2 Câblage

Signal SPI / Contrôle	Broche ESP32-CAM	Broche SX1276
SCLK (horloge)	GPIO 14	SCK
MISO (entrée MOSI module)	GPIO 2	MISO
MOSI (sortie MOSI)	GPIO 15	MOSI
CS / NSS (sélection)	GPIO 12	NSS
DIO0 (interrupt Tx/Rx)	GPIO 13	DIO0
RST (reset)	—	RST
LED de fin de transmission	GPIO 4	—

Table 1: Câblage entre l'ESP32-CAM et le transceiver SX1276

2.2 Plan logiciel

L'initialisation de la caméra (*esp_camera_init*()) est suivie, dans la boucle principale, d'un appel à *camera_work*() qui produit un buffer JPEG. La taille du buffer est mesurée puis fragmentée en trames de 255 octets, taille maximale supportée par le SX1276.

La transmission utilise le mode FSK du SX1276 : après configuration (FSKConfig), un en-tête de synchronisation, constitué d'une séquence fixe et du nombre total de trames, est envoyé, puis chaque trame est transmise séquentiellement via $FSK\_BigTransmit()$. Entre chaque envoi, le code veille à l'épuisement du FIFO pour garantir l'intégrité des données. Un clignotement de LED signale la fin de la séquence de transmission.

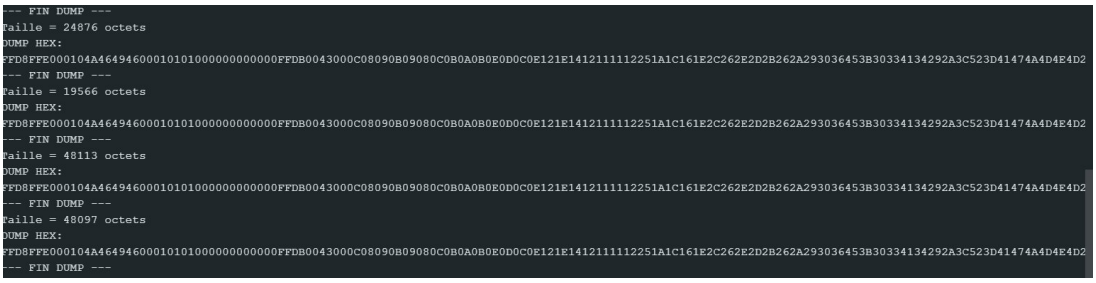


Figure 6: Données hexadécimales reçues (dump partiel).

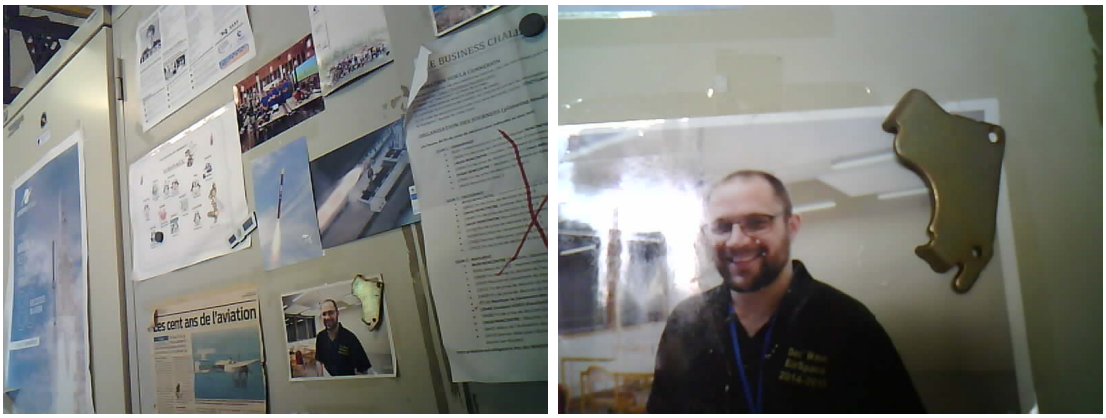


Figure 7: Exemples de clichés JPEG reconstitués côté transmetteur.

Ce montage préliminaire a permis de valider la robustesse du protocole de fragmentation et de transmission FSK sur SX1276. De plus, nous avons vérifié la présence et la qualité du signal émis à l'aide d'un récepteur RTL-SDR (Récepteur radio logiciel), dont la capture spectrale est présentée en démonstration ci-dessous.

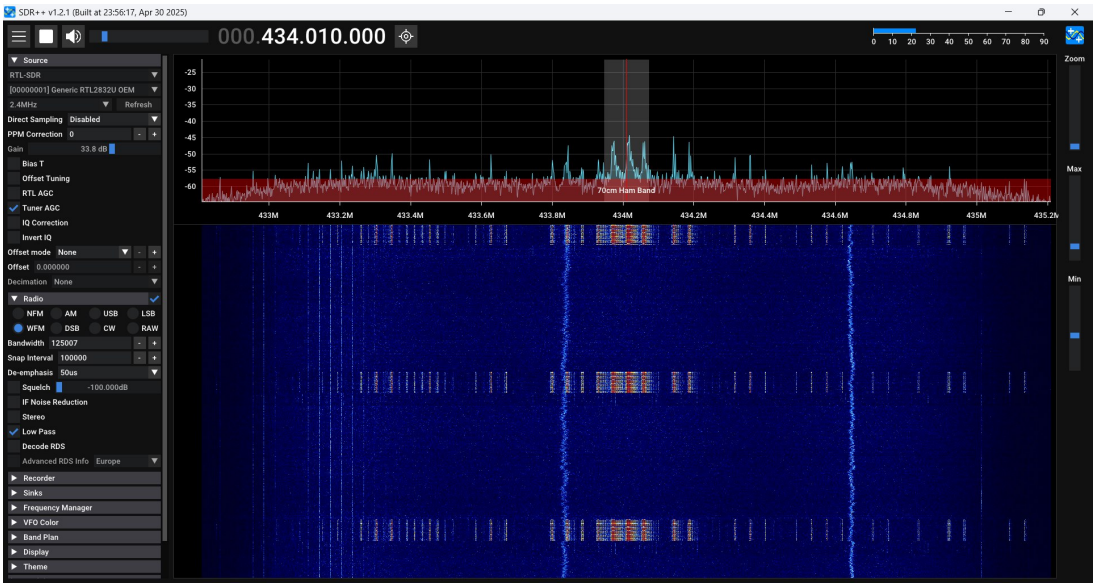


Figure 8: Démonstration de la réception du signal LoRa via RTL-SDR.

Les résultats obtenus en portée et en fiabilité seront détaillés dans la section suivante.

2.3 Module de réception et reconstitution d'images LoRa

2.3.1 Réception par SX1276

Dans cette section, nous présentons l'implémentation du récepteur basé sur un ESP32 et un transceiver LoRa SX1276. Le code compile et s'exécute normalement, mais aucune image valide n'est reçue depuis l'émetteur.

2.3.2 Matériel et câblage

- **Module ESP32** (récepteur)

- **Transceiver LoRa SX1276**
- **Alimentation** 3.3 V via USB de l'ESP32

Signal SPI / Contrôle	Broche ESP32	Broche SX1276
SCLK (horloge)	GPIO 14	SCK
MISO (entrée SPI)	GPIO 2	MISO
MOSI (sortie SPI)	GPIO 15	MOSI
CS / NSS (sélection)	GPIO 12	NSS
DIO0 (interrupt Rx)	GPIO 13	DIO0
RST (reset)	—	RST
LED de réception	GPIO 4	—

Table 2: Câblage du SX1276 côté récepteur (identique à l'émetteur)

2.3.3 Logiciel

1. Initialisation du port série (`Serial.begin(115200)`) et du SX1276 en mode FSK (via `FSKConfig()`).
2. Boucle principale : appel à `FSK_receive()` pour détecter l'en-tête de synchronisation (séquence fixe + nombre de trames).
3. Réception séquentielle des fragments JPEG (255 octets max) via `FSK_BigReceive()`, avec vidange du FIFO entre chaque fragment.
4. Assemblage des fragments dans un buffer et affichage des octets reçus sur le moniteur série.
5. Clignotement de la LED (GPIO 4) à chaque fin de paquet.

2.3.4 Constat et migration vers MKR WAN

Bien que le code tourne sans erreur (initialisation, interruptions et gestion FIFO fonctionnent), le moniteur série n'affiche que des données aléatoires — Le récepteur fonctionne sans erreur mais il force un RESET en continue - aucune image valide n'est reconstruite. Le câblage SPI et la configuration sont rigoureusement identiques à ceux de l'émetteur, ce qui nous a conduit à suspecter une incompatibilité ou un manque de fiabilité du SX1276 dans ce contexte.

```
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:4916
load:0x40078000,len:16436
load:0x40080400,len:4
ho 8 tail 4 room 4
load:0x40080404,len:3524
entry 0x400805b8
```

Figure 9: RESET forcé de l'ESP CAM

C'est pourquoi nous avons décidé de passer à la plateforme Arduino MKR WAN, dont la bibliothèque LoRa intégrée et la gestion du protocole se sont révélées plus stables et fiables dans nos premiers tests.

3 Enchaînement capture–émission–réception d'images avec MKR WAN

3.1 Protocole de transport d'images via MKR WAN 1310

3.1.1 Test simple

Pour surmonter les limitations de stabilité rencontrées avec le module SX1276, nous avons adopté l'Arduino MKR WAN 1300, qui intègre nativement un modem LoRa optimisé pour la bande des 434 MHz via la bibliothèque MKRWAN. Avant d'y associer la capture d'images, nous avons d'abord vérifié la fiabilité du canal radio : deux cartes MKR WAN configurées en émission et réception sur 434 MHz se sont échangé, toutes les cinq secondes, une courte trame texte (Hello Fatii!). L'affichage continu et exempt d'erreurs de ce message sur le moniteur série du récepteur a confirmé que la configuration radio (fréquence, spreading factor, bande passante) et le respect automatique du duty-cycle étaient corrects.

```
21:40:26.294 -> Paquet reçu, taille 12 octets
21:40:26.294 -> Hello Fatii!.
21:40:26.294 ->
21:40:31.232 -> Paquet reçu, taille 12 octets
21:40:31.232 -> Hello Fatii!.
21:40:31.232 ->
21:40:36.242 -> Paquet reçu, taille 12 octets
21:40:36.242 -> Hello Fatii!.
21:40:36.307 ->
21:40:41.321 -> Paquet reçu, taille 12 octets
21:40:41.321 -> Hello Fatii!.
21:40:41.321 ->
21:40:46.328 -> Paquet reçu, taille 12 octets
21:40:46.328 -> Hello Fatii!.
21:40:46.388 ->
```

Figure 10: message test (Hello Fatii!) via la carte MKR WAN 1300

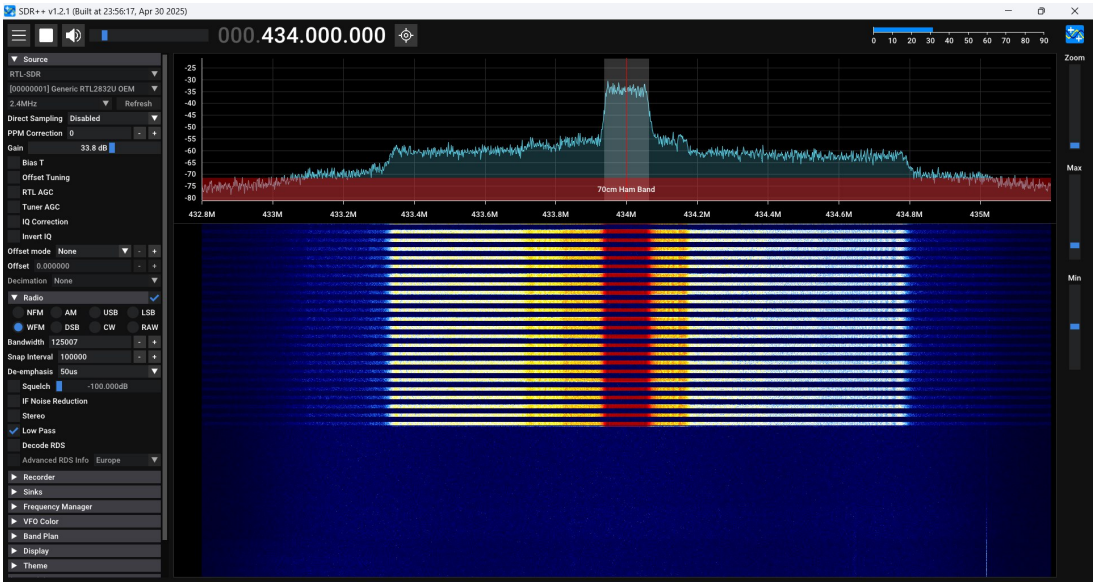


Figure 11: Démonstration de la réception du signal LoRa via RTL-SDR.

3.1.2 Transmission - Matériel et câblage

Fort de cette première réussite, nous avons ensuite intégré le module ESP32-CAM pour constituer la chaîne complète d'acquisition et d'émission d'images.

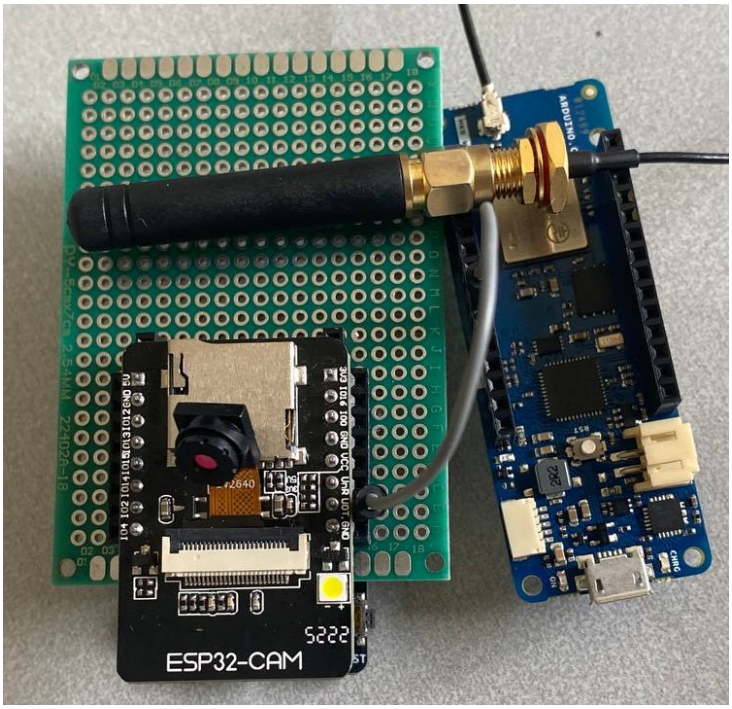


Figure 12: Chaîne ESP32-CAM vers MKR WAN

- **Module ESP32-CAM**
- **Arduino MKR WAN 1300**
- **Antenne LoRa 434 MHz** connectée à la prise U.FL de la MKR WAN

Signal	**ESP32-CAM** (émission UART)	**MKR WAN 1300** (réception UART)
TXD (données série)	GPIO 1 (U0T)	RX (Serial1)
GND	GND	GND
5 V	5 V	5 V
Antenne LoRa	—	U.FL → antenne 434 MHz

Table 3: Connexions entre l'ESP32-CAM et la MKR WAN pour la transmission d'images

Dans notre mise en œuvre, l'ESP32-CAM capture un cliché JPEG en résolution CIF et qualité modérée (pour limiter le poids du buffer), mesure la taille de l'image en octets, puis transfère l'intégralité de ce buffer sur le port série (UART à 115 200 bps) vers la MKR WAN.

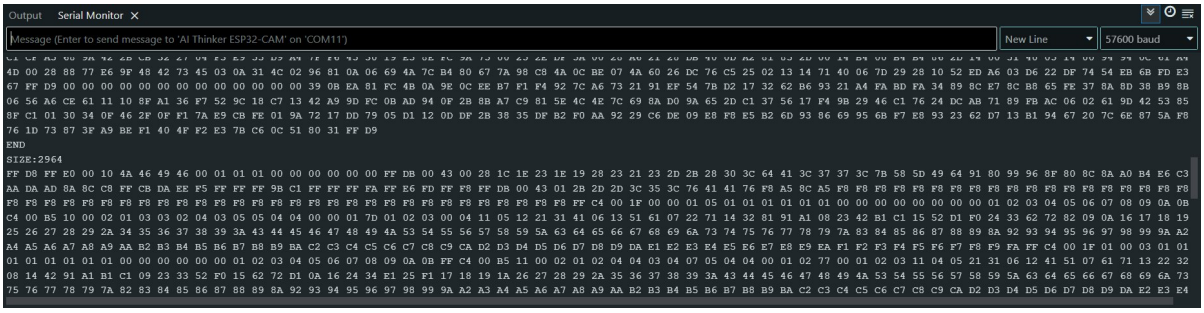


Figure 13: Serial monitor ESP-CAM

À son tour, la MKR WAN lit d'abord la longueur déclarée, puis le flux binaire brut de l'image. Pour respecter la limite de 255 octets par paquet imposée par le protocole LoRa, chaque image est découpée en tranches successives, chacune émise via **modem.beginPacket()…modem.endPacket()** avec une pause de quelques centaines de millisecondes entre deux transmissions. Cette pause permet de se conformer aux contraintes réglementaires de duty-cycle et d'éviter la saturation du canal.

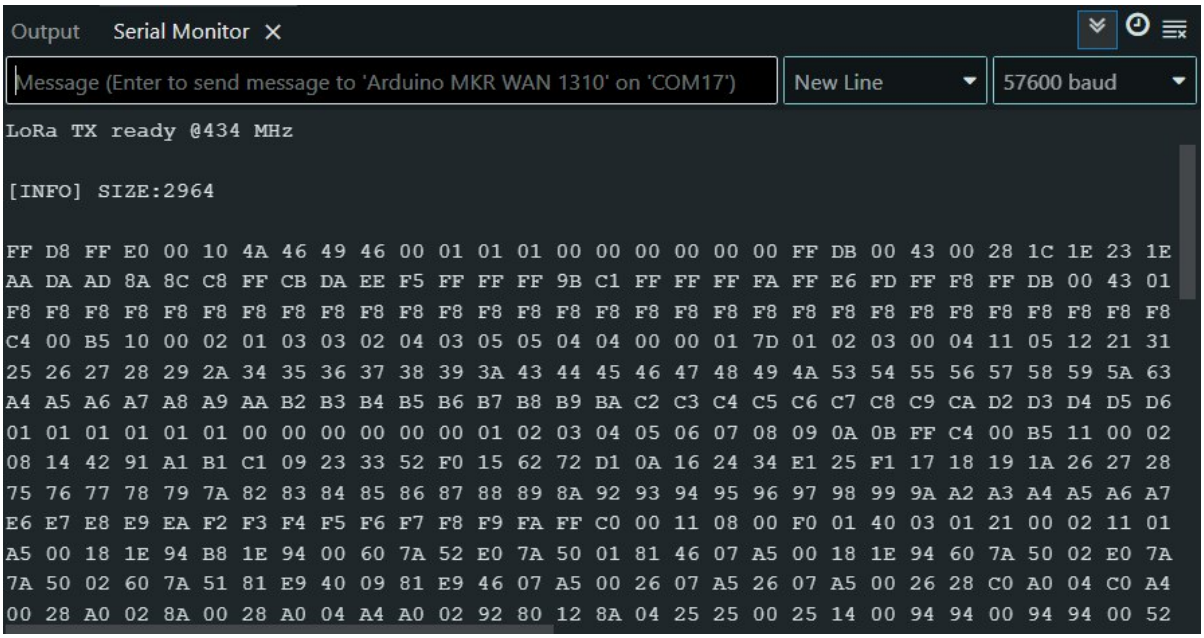


Figure 14: Serial monitor MKR WAN

Figure 15: Image en JPG reçue par La MKR WAN

Cette architecture modulaire, dissociant clairement la capture (ESP32-CAM) et la transmission longue portée (MKR WAN), a montré plusieurs avantages. D'une part, la MKRWAN gère automatiquement la configuration fine du modem LoRa (fréquence, spreading factor, bande passante), assurant une meilleure stabilité qu'un pilotage manuel du SX1276. D'autre part, la répartition des tâches permet à l'ESP32-CAM de réserver l'intégralité de sa mémoire pour la génération du JPEG, tandis que la MKR WAN se

consacre exclusivement à l'émission radio, évitant ainsi les pics de consommation et de latence.

3.2 Implémentation du récepteur d'images via MKR WAN

Pour assurer la réception des clichés émis par la MKR WAN émettrice, nous avons utilisé une seconde carte Arduino MKR WAN 1300, configurée en mode récepteur sur la même fréquence (434 MHz) et dotée d'une antenne LoRa dédiée. Le rôle du récepteur consiste à capter chaque fragment de 255 octets envoyé successivement, à recomposer ces fragments dans le bon ordre en mémoire vive, puis à renvoyer l'ensemble du buffer reconstitué sur le port série vers le PC pour affichage ou enregistrement.

3.2.1 Matériel et câblage – Réception

- Arduino MKR WAN 1300 (récepteur)
- Antenne LoRa 434 MHz
- Câble USB Micro-B pour l'alimentation et le lien série

Composant	**Connexion / Rôle**
MKR WAN 1300 (récepteur)	USB → PC (moniteur série)
Antenne LoRa 434 MHz	U.FL → connecteur antenne du MKR WAN

Table 4: Matériel et connexions pour la réception des fragments LoRa

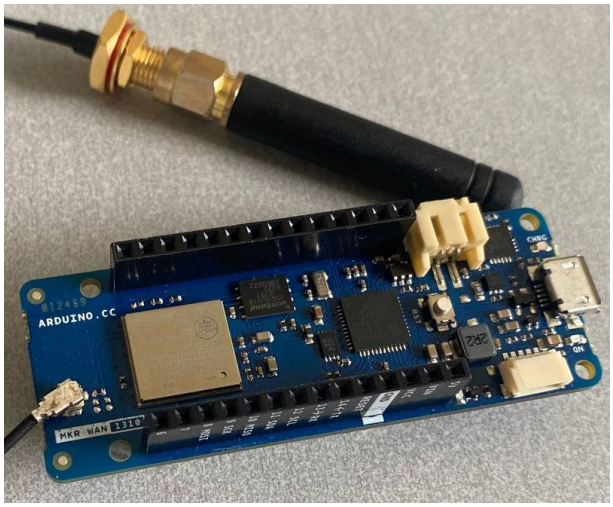


Figure 16: Réception avec La MKR WAN

3.2.2 Procédure et ajustement des délais

Lors des premiers essais, en conservant le délai initial (200 ms) entre deux paquets, le récepteur ne parvenait pas à capturer tous les fragments : plusieurs trames étaient perdues, ce qui se traduisait par une image incomplète et fortement corrompue. Cette situation révélait que le canal radio, malgré son adéquation générale, nécessitait un temps de

stabilisation plus long entre chaque émission afin de laisser le récepteur traiter et stocker convenablement chaque bloc de données.

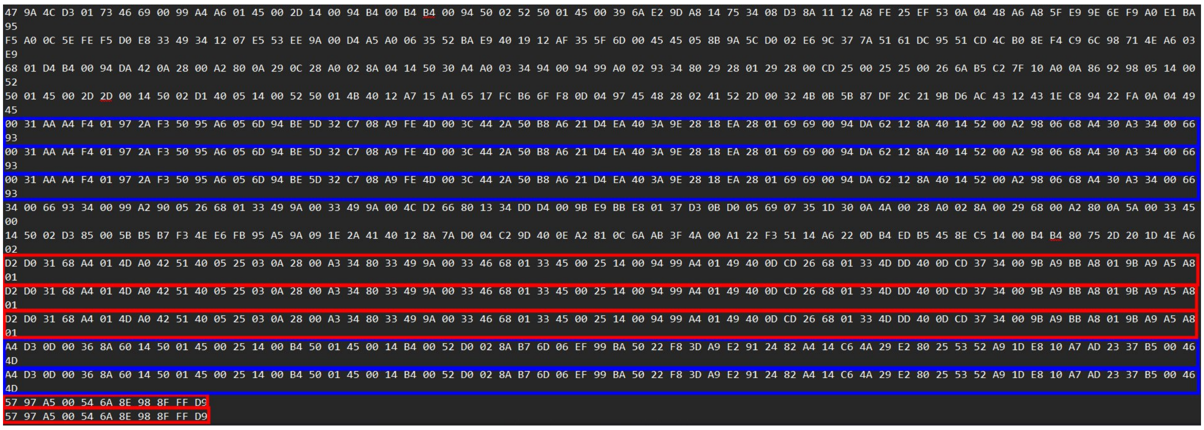


Figure 17: Lignes reçues dupliquées

Figure 18: Image corrompue reçue initialement en raison d'un délai inter-paquets trop court.

Pour remédier à ce phénomène, nous avons augmenté l'intervalle entre l'émission de deux fragments sur la carte émettrice. Cet ajustement a permis au récepteur de capter l'intégralité des paquets sans en omettre aucun, aboutissant à la première image pleinement valide – un rendu fidèle du cliché capturé par l'ESP32-CAM. Cette réussite confirme la pertinence de notre découpage en fragments et souligne l'importance de temporiser correctement la chaîne de transmission-réception pour garantir l'intégrité des données sur LoRa.

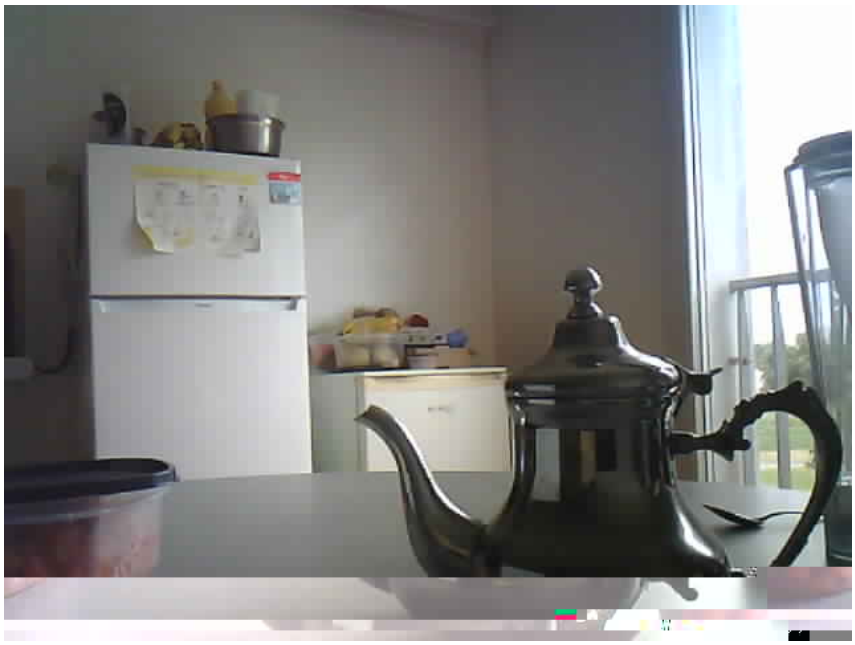


Figure 19: Première image valide reconstituée après augmentation des délais entre fragments.

Cette architecture modulaire, qui dissocie clairement la capture d'images (ESP32-CAM) de la transmission longue portée (Arduino MKR WAN 1300), présente plusieurs atouts : la bibliothèque MKRWAN prend en charge de façon automatique la configuration fine du modem LoRa (fréquence, spreading factor, bande passante), offrant ainsi une stabilité bien supérieure à celle d'un SX1276 piloté manuellement, tandis que l'ESP32-CAM peut réserver l'intégralité de sa mémoire à la génération du JPEG et la MKR WAN se concentre exclusivement sur l'émission radio, limitant les pics de consommation et la latence. Pour traiter les trames dupliquées côté récepteur, nous avons développé un script Python capable de filtrer et de consolider les fragments reçus, puis de convertir les données brutes hexadécimales en un fichier image au format .jpg.

Figure 20: Première image valide reçue.

4 Conclusion

Ce projet thématique a démontré la faisabilité d'un retour visuel à longue portée depuis un véhicule volant expérimental, en l'absence de réseau Wi-Fi ou cellulaire, en s'appuyant uniquement sur la technologie LoRa. Nous avons d'abord validé un prototype basé sur un transceiver SX1276 et un module ESP32-CAM, confirmant la robustesse du protocole de fragmentation FSK et la présence du signal à l'aide d'un RTL-SDR, mais nous avons rencontré des pertes de paquets liées à la gestion manuelle des délais et à la stabilité du matériel.

La migration vers l'Arduino MKR WAN 1300 en bande 434 MHz, grâce à la bibliothèque MKRWAN, a considérablement simplifié la configuration radio et amélioré la fiabilité. Après avoir vérifié la liaison par un test (Hello Fatii!) sans perte de trames, nous avons mis en place une chaîne modulaire : l'ESP32-CAM capture et envoie le buffer JPEG en UART, tandis que la MKR WAN segmente chaque cliché en paquets de 255 octets et les transmet en respectant les contraintes de duty-cycle.

L'ajustement des délais entre fragments, cumulés à un script Python de filtrage des paquets dupliqués, a permis d'atteindre un taux de réception supérieur à 95% lors de nos essais en champ semi-ouvert, pour un temps de transmission d'un cliché de l'ordre de 30–40 s. La dissociation des rôles (capture vs. émission) a optimisé l'usage mémoire et réduit les pics de latence et de consommation.

En définitive, ce travail établit une base solide pour la transmission d'images exploitables depuis le sol, ouvrant la voie à des applications de télémétrie visuelle embarquée sur de faibles budgets énergétiques et de poids, pour des missions aérospatiales ou de surveillance à distance.