

Projet prospection implantation d'éolienne Char Terrestre

GRUBER NOÉ

COSSON KILLIAN

BEAL JULIEN



force
dimension

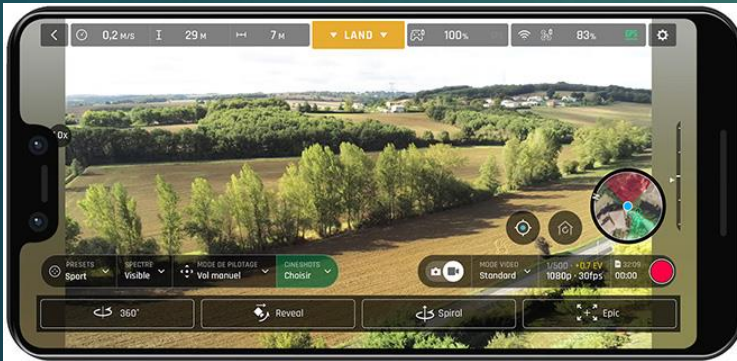
EEF SAS
est une entreprise du groupe
eno energy GmbH

 **Cité Scolaire
BRIFFAUT**

Sommaire

- ▶ Mise en contexte
- ▶ Diagramme exigence
- ▶ Présentation tâche n°1
 - ▶ Commande du char
- ▶ Présentation tâche n°2
 - ▶ Télémétrie
- ▶ Présentation tâche n°3
 - ▶ Gestion vidéo

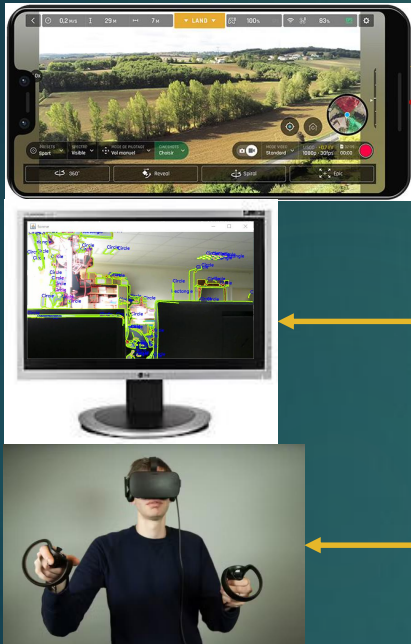
Mise en contexte



Mise en contexte : interactions entre les différents modules

- ▶ Commande
- ▶ télémétrie
- ▶ Vidéo

Utilisateur



Robot haptique

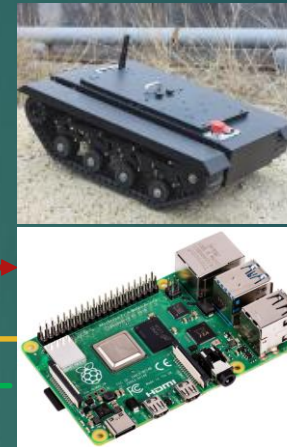


Serveurs :

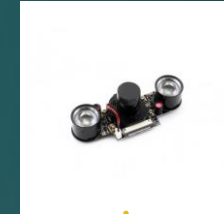
- Pilotage
- Télémétrie
- Vidéo



Char



Vidéo



Capteurs



Moteurs



Diagramme Exigence

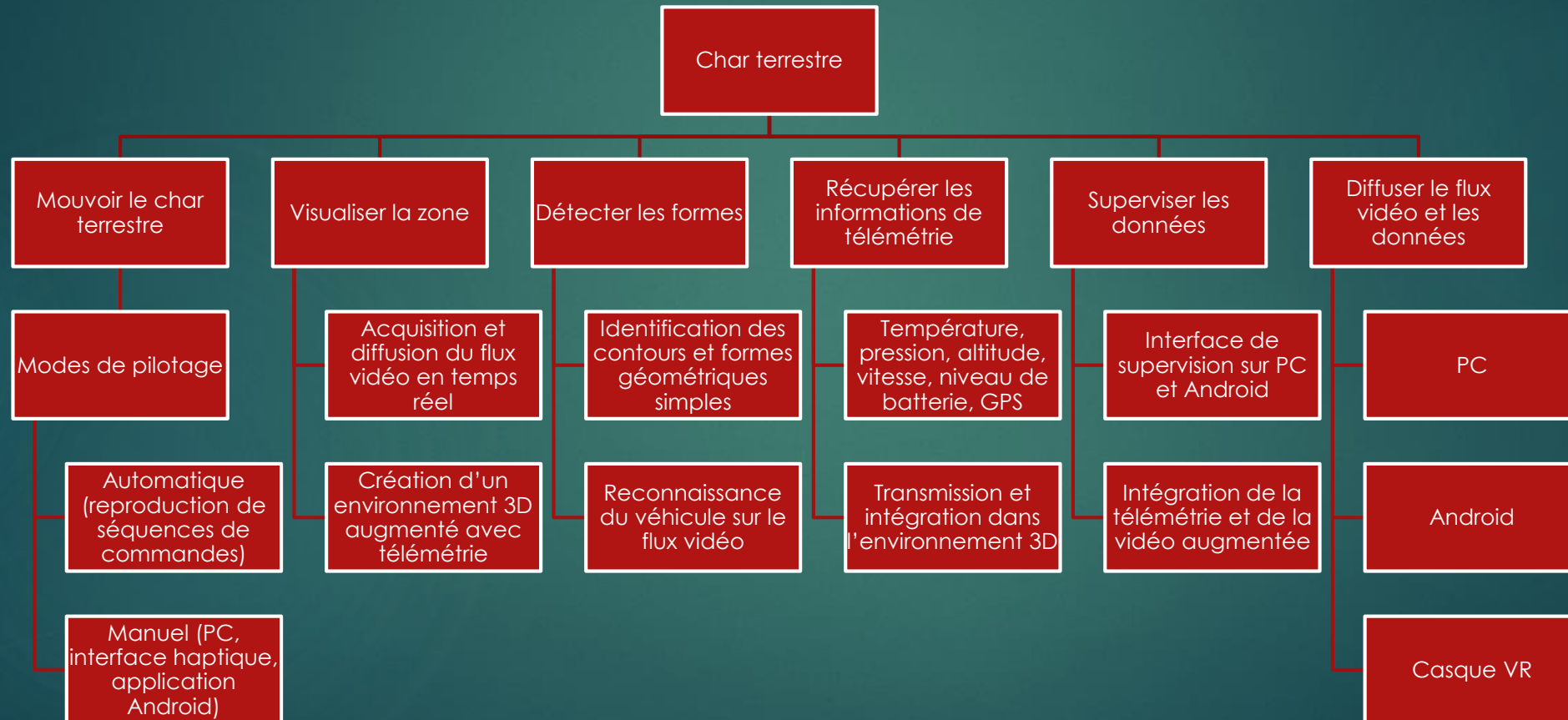
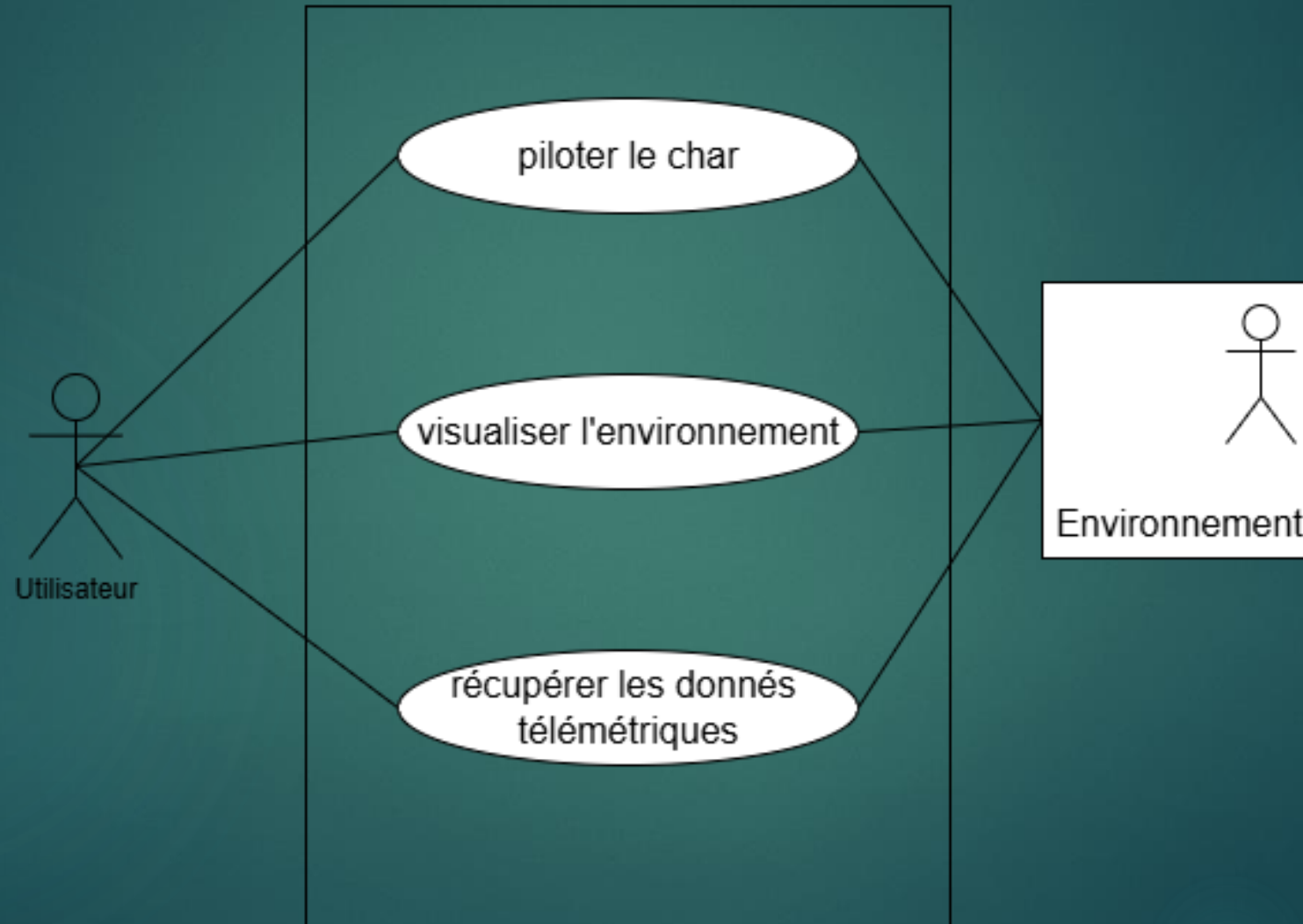



Diagramme des cas d'utilisation





Présentation de tache n°1

DRONE CHAR TERRESTRE

ETUDIANT N°1

GRUBER NOÉ

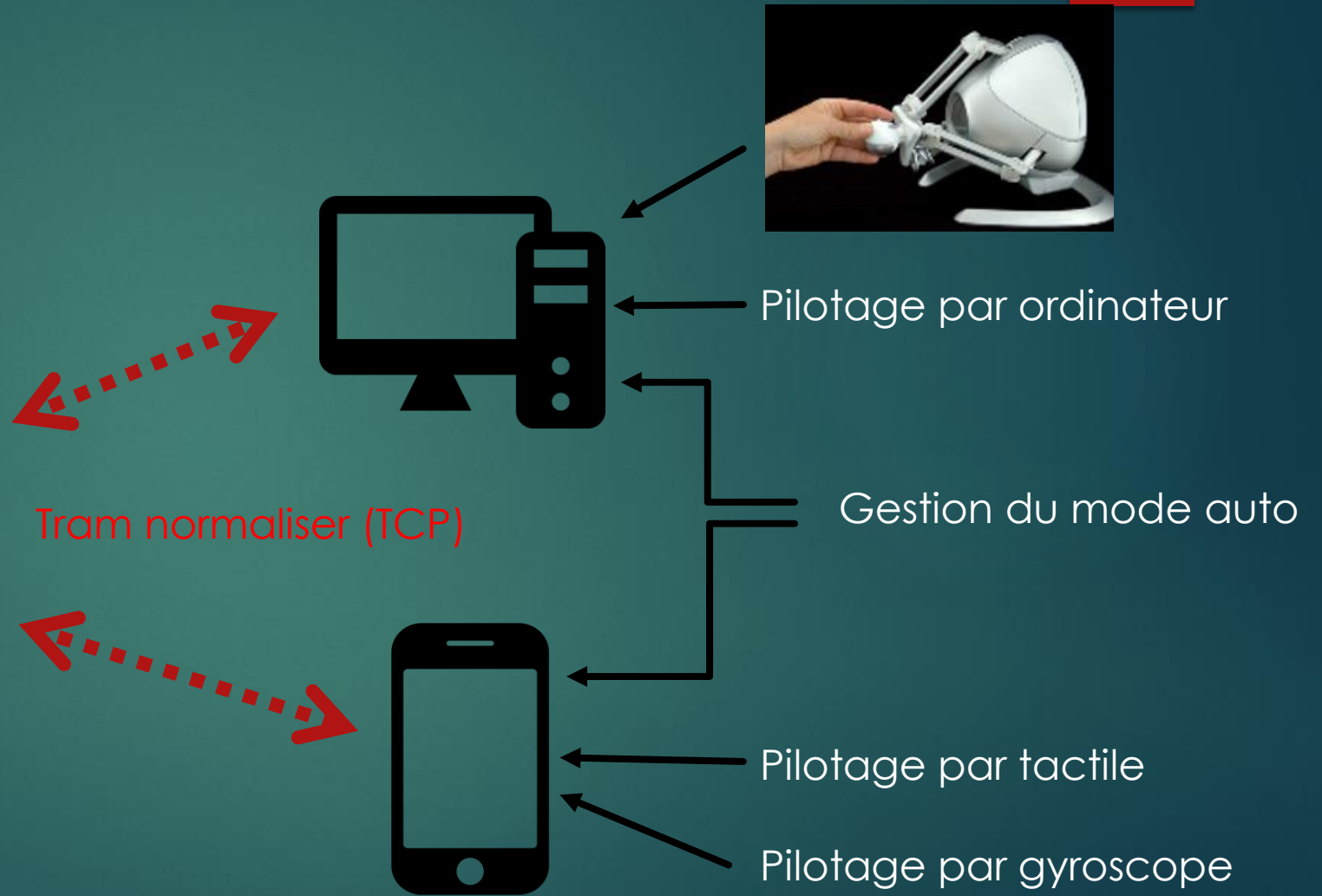
Sommaire

- ▶ Présentation de la tâche
- ▶ Contrôle des moteurs
- ▶ Réception, traitement des trames
- ▶ Pilotage Manuel
- ▶ Pilotage automatique

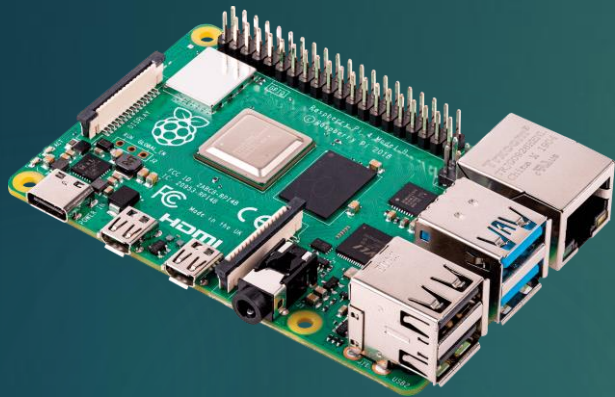
Présentation de la tâche



Présentation de la tâche



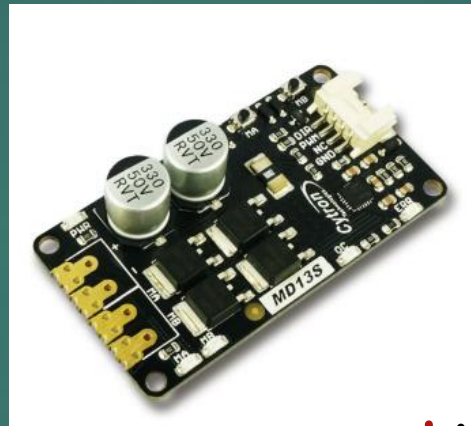
Contrôle des moteurs



1 Pin
Pour le sens
1 Pin GND



1 Pin PWM



25% Width Pulse



50% Width Pulse



75% Width Pulse



Réception, traitement des trames

- ▶ Trame gestion système : « Cmd : ShutDown », « Cmd : ClientDeConnecte ».

```
Attente de la Requete Client...
Envoi de 1 accuse de reception au Client

Cmd : ShutDown

^ La Commande ci-dessus a ete recue ^

Fermeture du serveur par le Client
la vitesse du moteur gauche est : 0.0 ; et du moteur droit est : 0.0
```

- ▶ Trame de pilotage : « |X/Y| » ou X et Y sont la position des Joystick.

```
|17/44|

^ La Trame de Donnees ci-dessus a ete recue ^

le joystick et en x: 17.0 ; y: 44.0

la vitesse du moteur gauche est : 61.0 ; et du moteur droit est : 27.0
```

- ▶ Trame de mode auto : « Dis:X/Y/Z| » ou X et la longueur, Y la largeur du terrain et Z le nombre de passage.

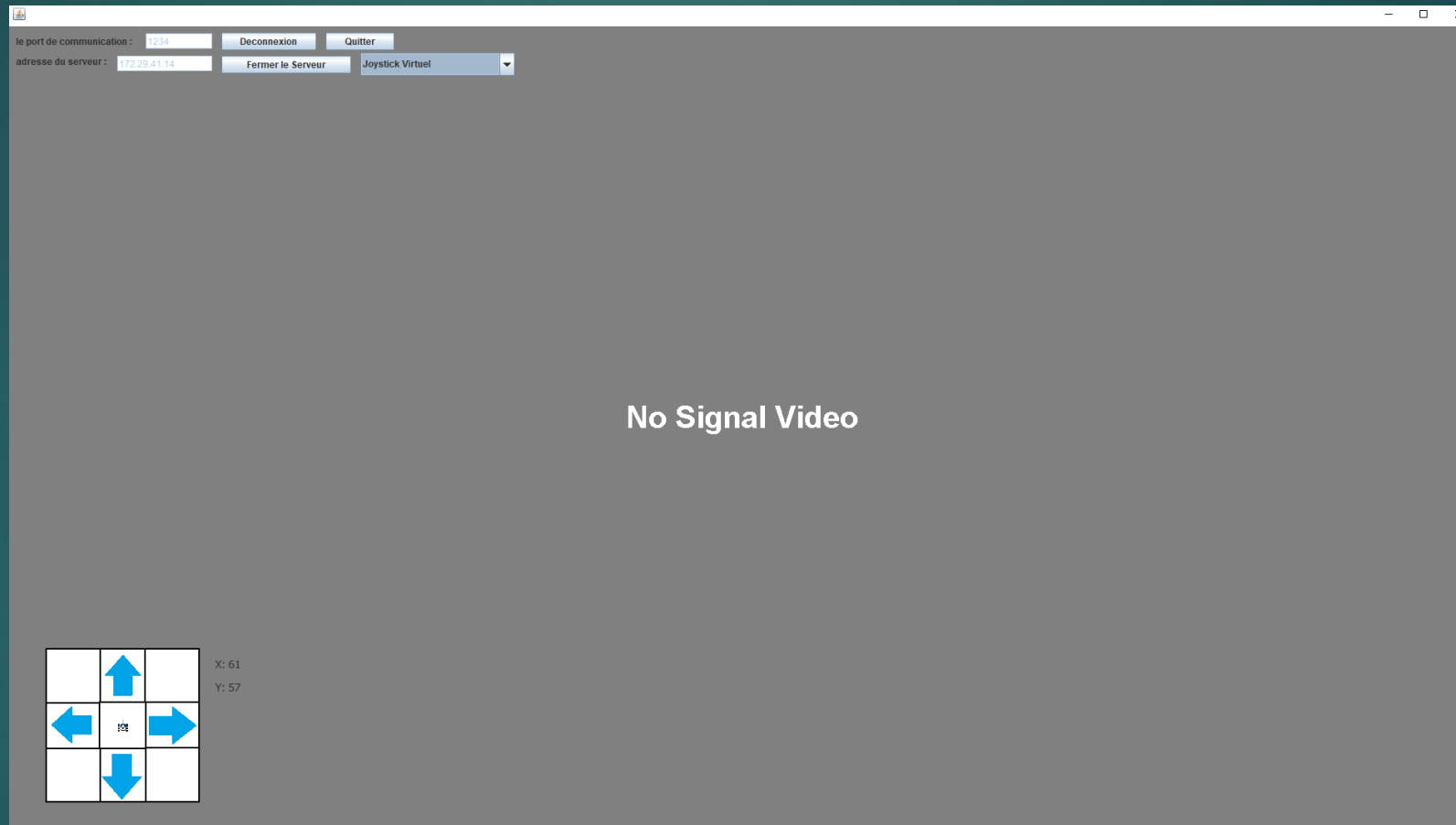
```
Dis:10/5/2|

^ La Trame de Donnees ci-dessus a ete recue ^

le char vas patrouiller sur une Longueur de : 10.0 m,
une Largeur de : 5.0 m,
il fera 3 pasage,

Attente de la Requete Client...
la vitesse du moteur gauche est : 75.0 ; et du moteur droit est : 75.0
```

pilotage manuel par PC



Type de frame « | X/Y | » envoyer

Pilotage manuel par robot haptique

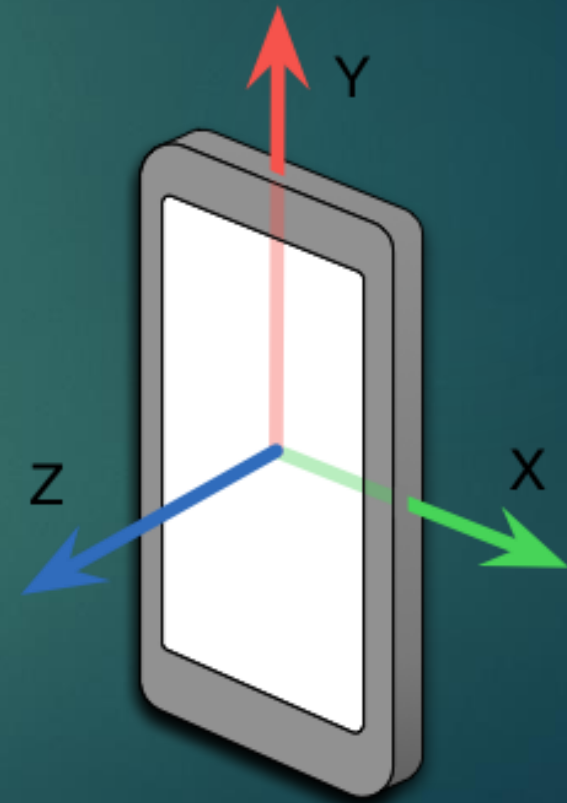


Normalisation
pour envoi



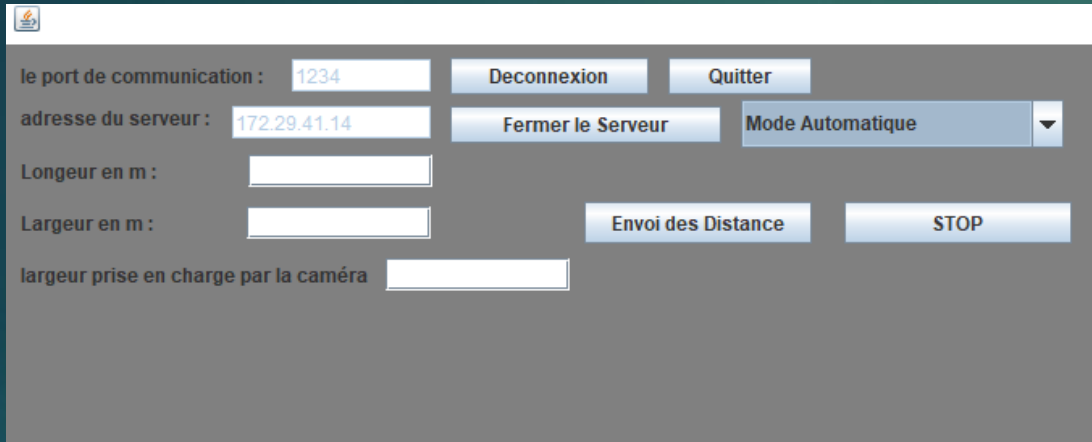
Type de frame « | X/Y | » envoyer

Pilotage par application Android



Type de frame « | X/Y | » envoyer

Pilotage automatique



le port de communication :

adresse du serveur :

Longeur en m :

Largeur en m :

largeur prise en charge par la caméra

Type de frame « Dis:X/Y/Z | » envoyer



Modules accéléromètres, gyroscopes : MPU6050

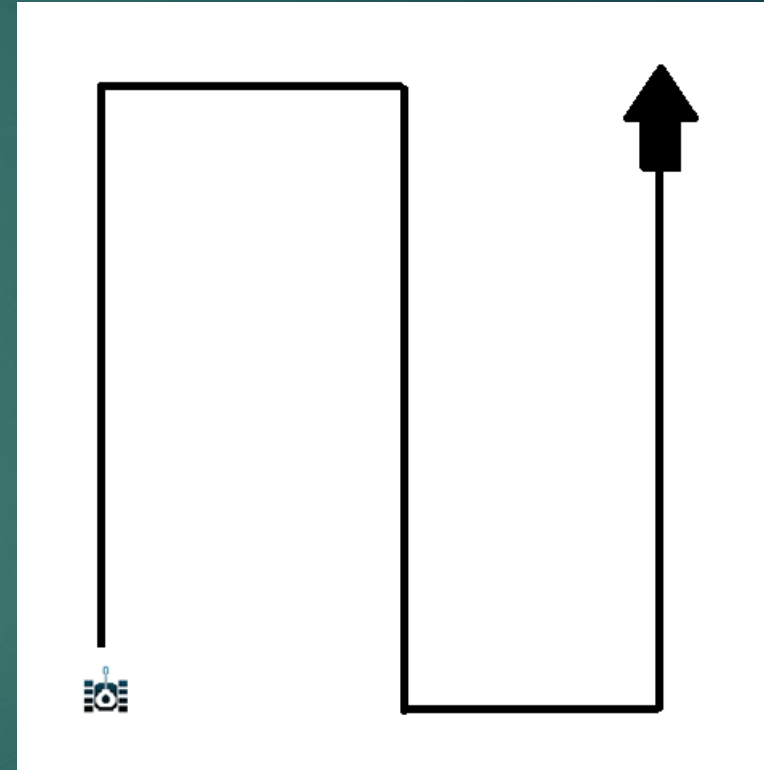
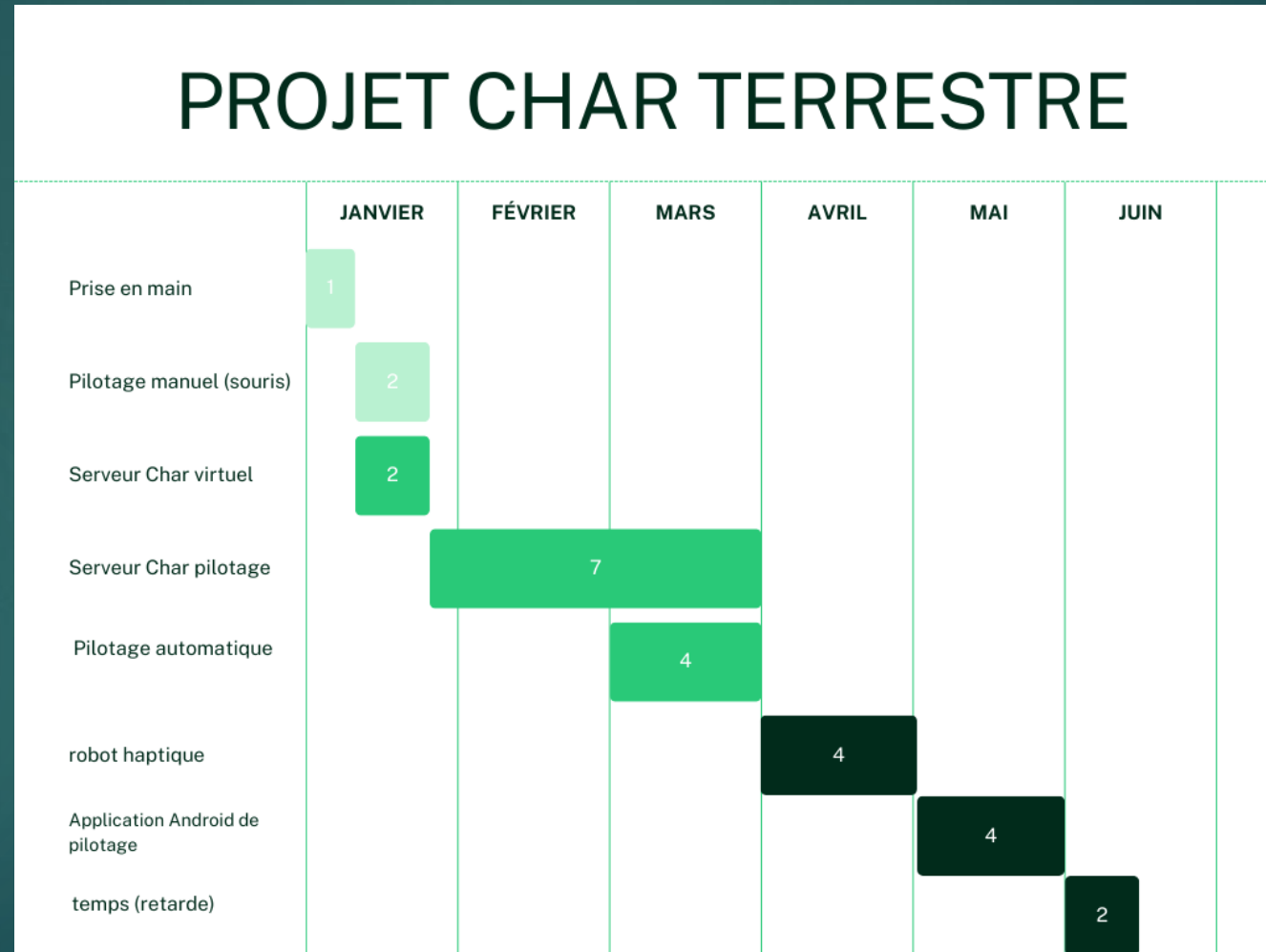



Diagramme de Gantt





Présentation de tache n°2

DRONE CHAR TERRESTRE

ETUDIANT N°2

COSSON KILLIAN

Présentation de la tâche

- ▶ Données télémétrique
- ▶ connexion client/serveur
- ▶ Raspberry pi
- ▶ Application Android
- ▶ Diagramme Gantt

Données Télémétrique

- ▶ Télémétrie du char application Java de supervision
- ▶ Télémétrie du char application Android de pilotage manuel

```
//trame pour le moment  
"dist:"+ v15310x.getDistance() +  
" ;templ:"+ bmp280.getTemperature() +  
" ;temph:"+ bmp280.getTemperature() +  
" ;alt:"+ bmp280.getAltitude() +  
" ;h:"+ rng() +  
" ;bat:"+ rng() +  
" ;baro:"+ bmp280.getPression() +
```

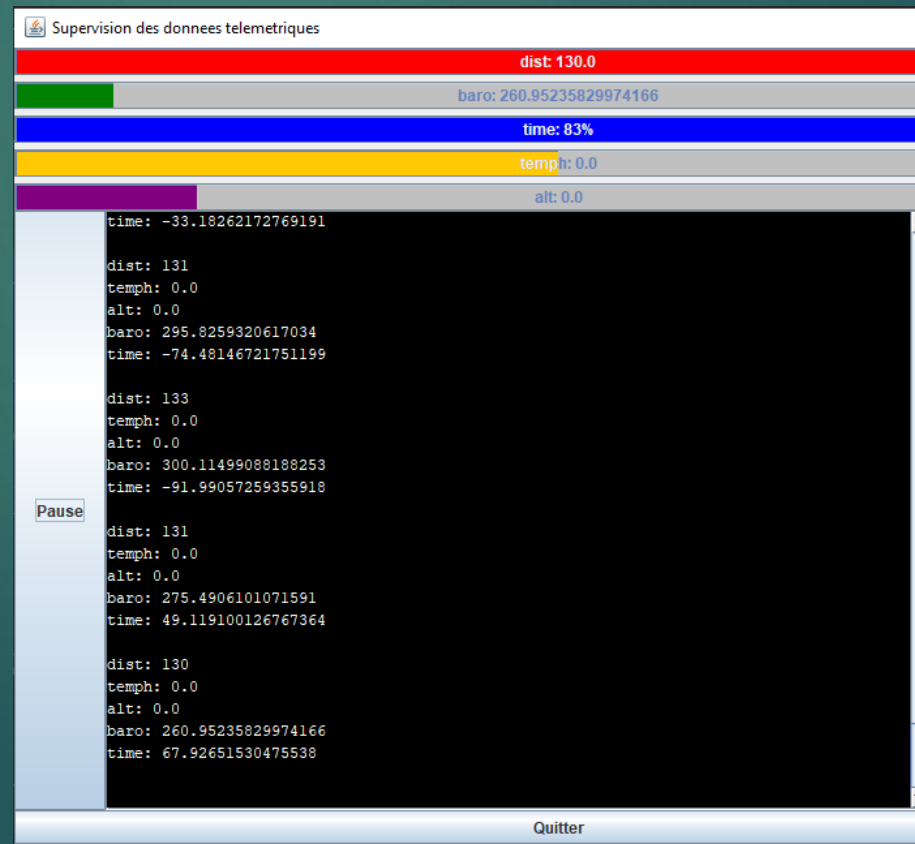
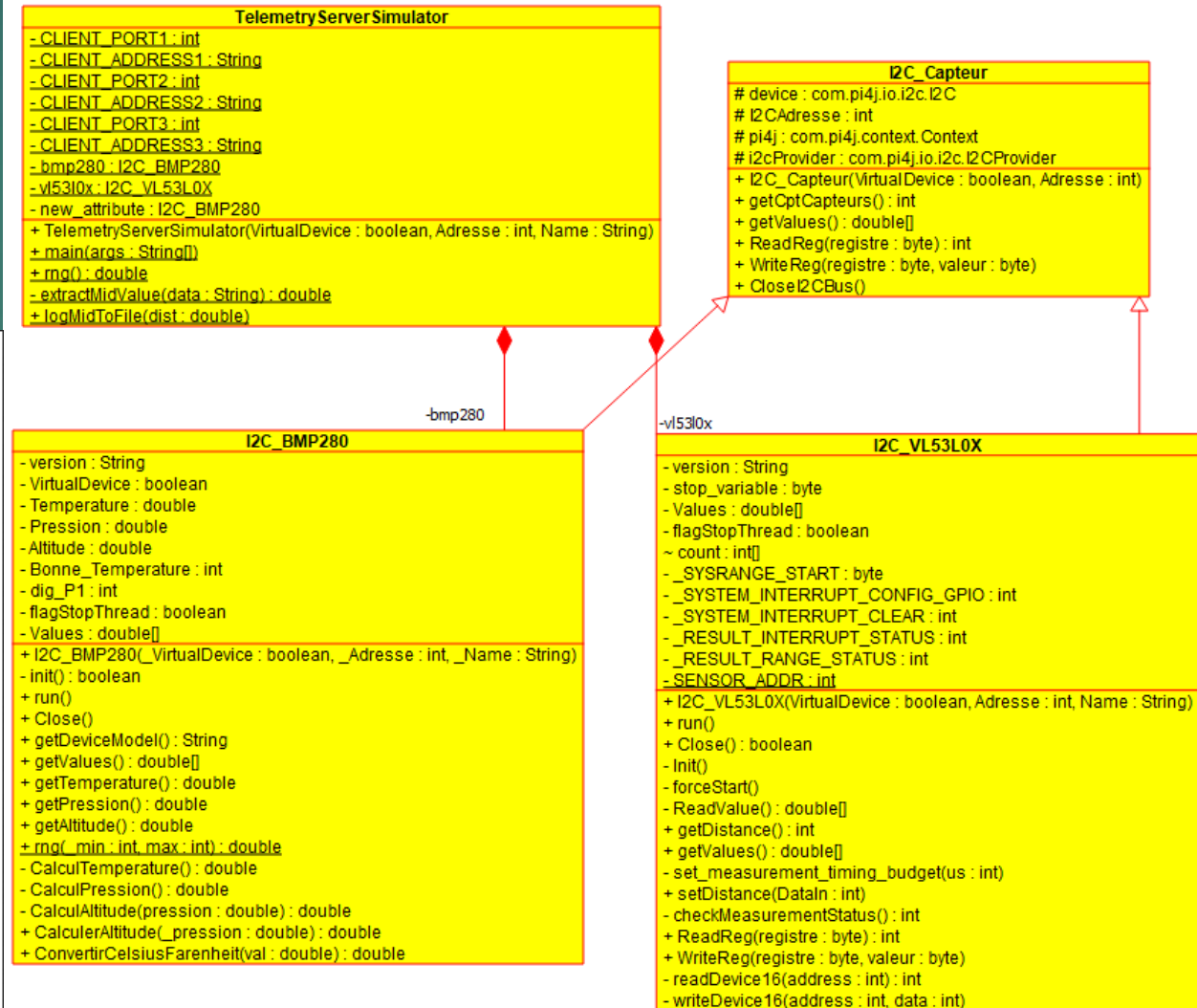
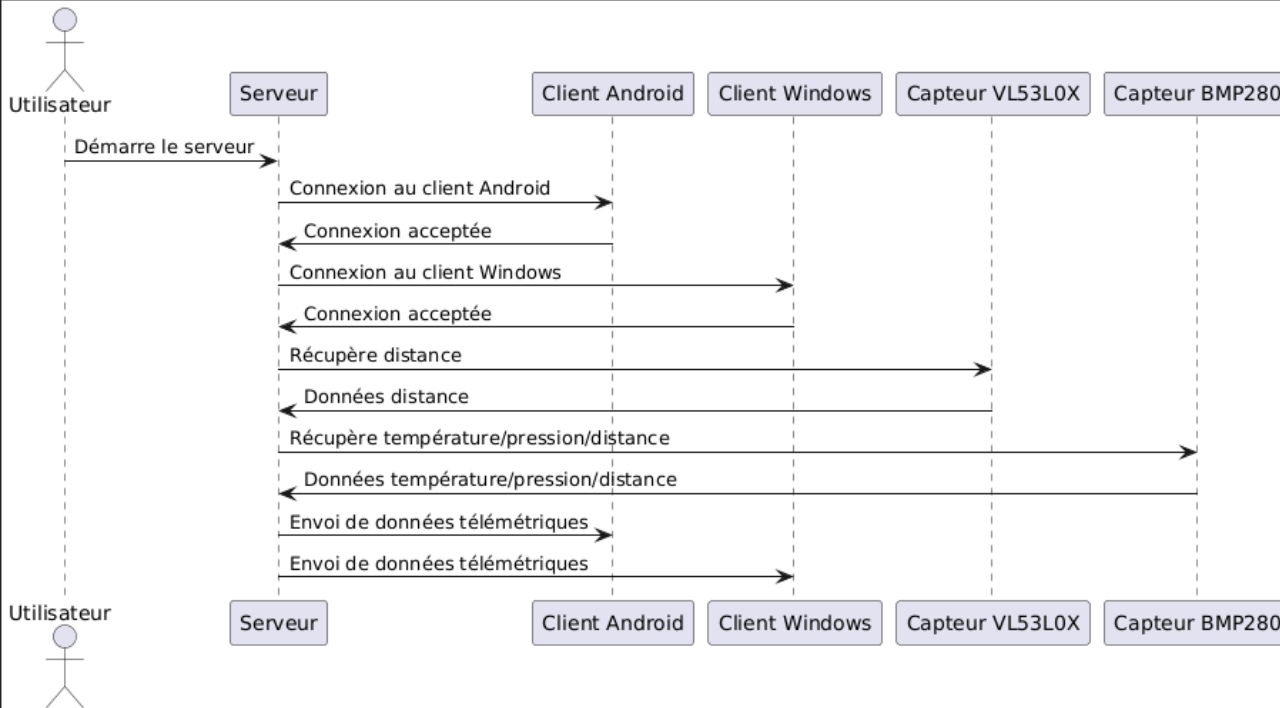


Diagramme actuel de données téléométrique



Java connexion client/serveur UDP + télémétrie

- ▶ Client java UDP de télémétrie
- ▶ Serveur UDP de télémétrie Hébergé par un raspberry

```
// Convertit la chaîne en octets avec l'encodage par défaut du système
byte[] sendData = data.getBytes();

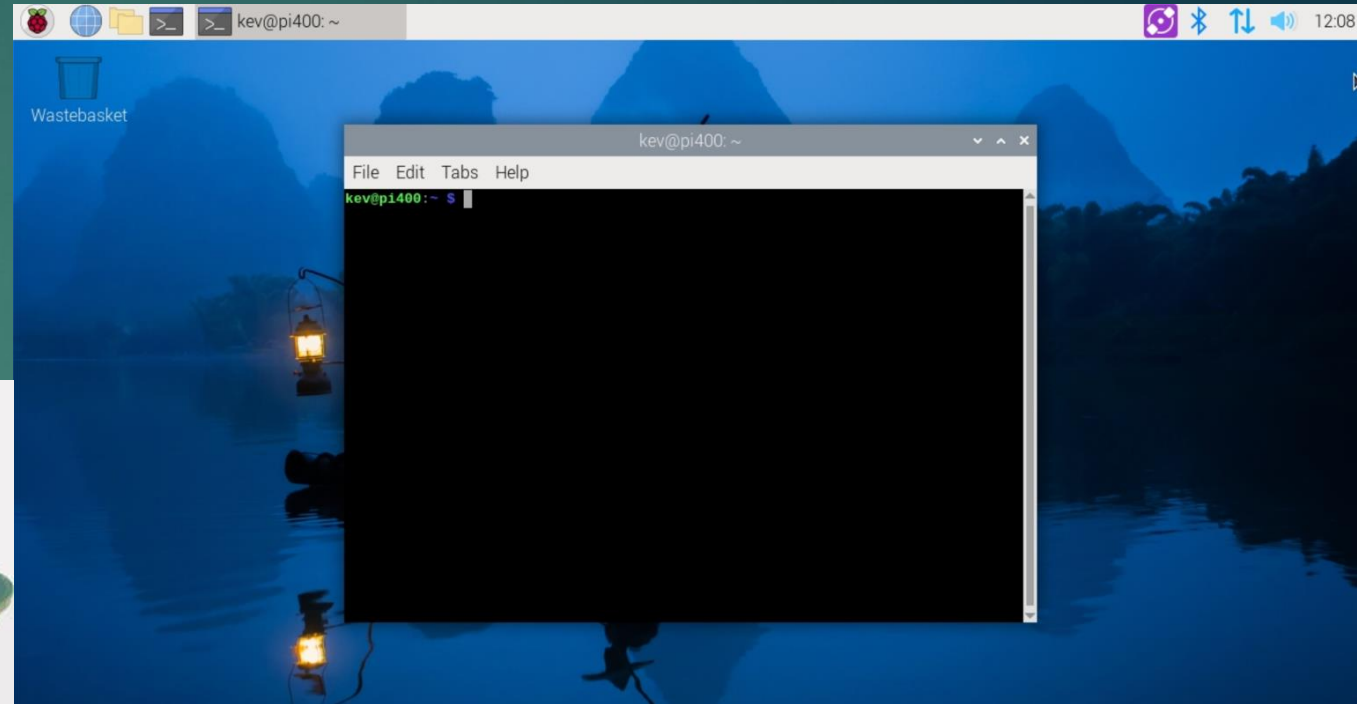
// Creation et envoi du paquet
for (int i = 0; i < clientAddresses.length; i++)
{
    DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, clientAddresses[i], clientsPorts[i]);
    serverSocket.send(sendPacket);
}

System.out.println("Donnees envoyees : " + data);

// Appel de la méthode pour recevoir et décoder les données
String receivedData = receiveAndDecodeData(socket, buffer);
```

Raspberry pi

- ▶ Raspberry, serveur du char
- ▶ Samba

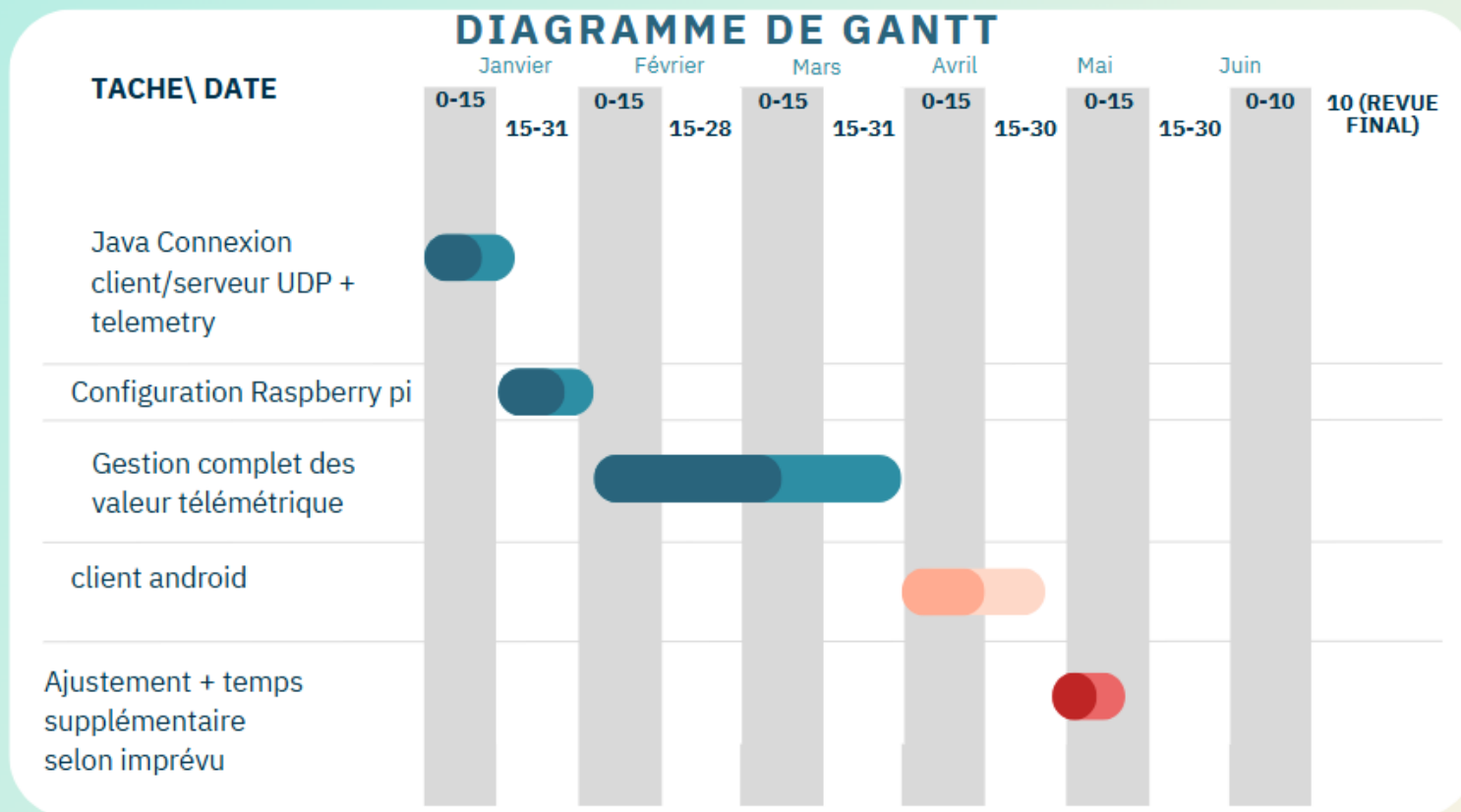



Application Android, et valeurs télémétrique

Application qui récupère les valeurs télémétrique



Système de prospection terrestre - Drones





Présentation de tache n°3

DRONE CHAR TERRESTRE

ETUDIANT N°3

BEAL JULIEN

Présentation de la tâche

Sommaire :

- ▶ Traitement du flux vidéo :
 - ▶ Chaîne d'information
 - ▶ Méthodes d'envoi et de réception vidéo
 - ▶ Détecter les contours
 - ▶ Détecter les formes
- ▶ Élaborer un environnement 3d :
 - ▶ Intégrer la vidéo
 - ▶ Intégrer la télémétrie du char
- ▶ Diffusion du flux vidéo augmenté :
 - ▶ PC
 - ▶ Android
 - ▶ VR

Chaine d'information



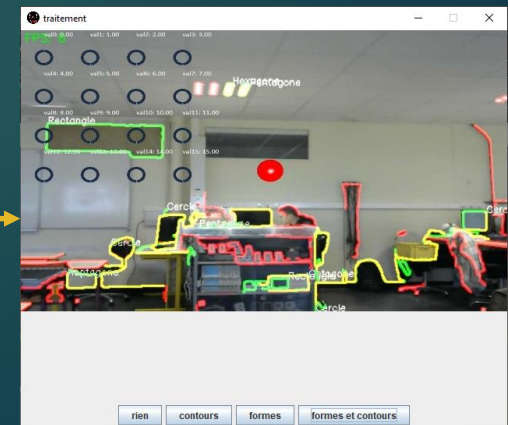
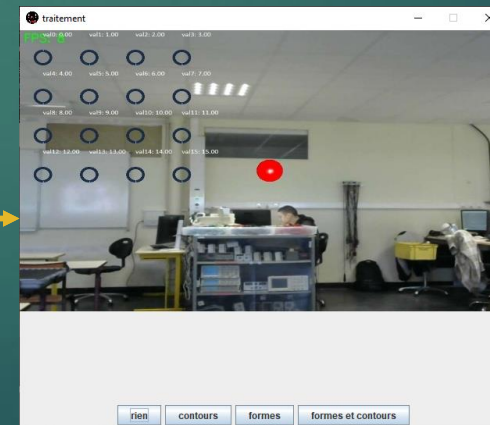
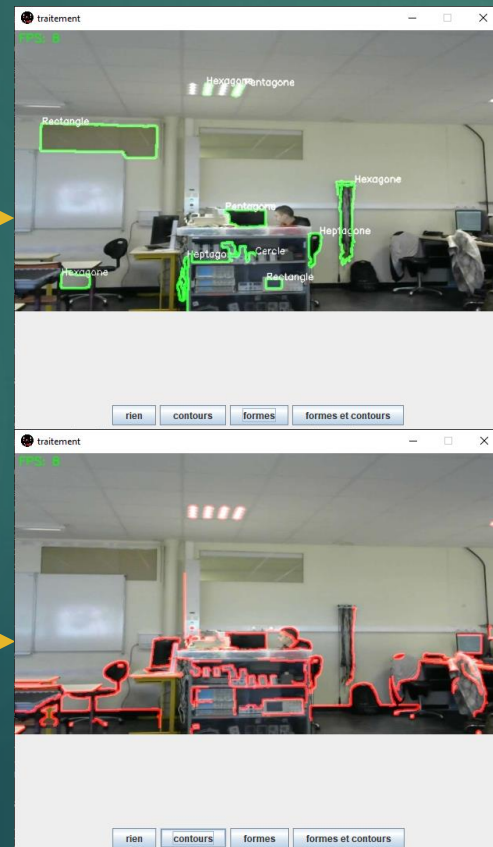
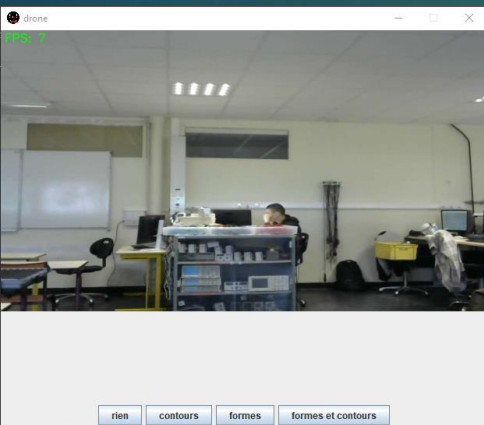
Image d'entrée



Détections / ajout télémétrie



Image de sorti



Créer méthodes d'envoi et de réception vidéo

- ▶ Envoie de l'image depuis la camera du char
- ▶ Réception + traitement de l'image
- ▶ Réception + Affichage de l'image traitée à utilisateur



+



Diagramme de séquence de communication

► Drone

- Démarrage du drone
- Boucle
 - Récupération de la vidéo
 - Si serveur connecté :
 - Sinon : Envoie du flux vidéo au serveur
 - Aucun envoie



► Serveur (traitement)

- Démarrage du serveur
- Boucle
 - Envoie donné au drone
 - Réception du flux vidéo
 - Si flux vidéo reçu :
 - Traitement du flux vidéo
 - Affichage du flux vidéo
 - Si client connecté :
 - Envoie du flux vidéo au client



► Client

- Démarrage du client
- Boucle
 - Envoie donné au serveur
 - Si flux vidéo reçu :
 - Affichage du flux vidéo



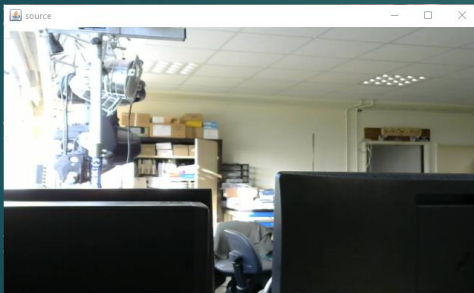
Traitement du flux vidéo

► Détection de contours

► Vidéo reçu :

► Traitement :

► Vidéo traiter :



Fonction de traitement du flux vidéo

Code :

```
while (true) {
    if (detection == true) {
        // Réduire la taille de l'image avant le traitement
        Mat resizedFrame = new Mat();
        Size reducedSize = new Size(frame.width() / 2, frame.height() / 2);
        Imgproc.resize(frame, resizedFrame, reducedSize);

        // Initialiser edgesRed avec la taille réduite
        edgesRed = Mat.zeros(resizedFrame.size(), resizedFrame.type());

        // Convertir l'image réduite en niveaux de gris
        Imgproc.cvtColor(resizedFrame, grayImage, Imgproc.COLOR_BGR2GRAY);

        // Appliquer un filtre bilatéral (lissage tout en préservant les contours)
        Imgproc.bilateralFilter(grayImage, flouImage, 9, 75, 75);

        // Détection des contours avec l'algorithme Canny
        int lowerThreshold = 50;
        int upperThreshold = 200;
        Imgproc.Canny(flouImage, edges, lowerThreshold, upperThreshold);

        // Appliquer la dilatation pour augmenter la taille des contours
        Mat kernel = Imgproc.getStructuringElement(Imgproc.MORPH_RECT, new Size(3, 3)); // Taille 3x3
        Imgproc.dilate(edges, dilatedEdges, kernel);

        // Appliquer la couleur rouge (vectorisé)
        edgesRed.setTo(new Scalar(0, 0, 255), dilatedEdges); // Affecter rouge là où les contours sont présents

        // Réagrandir l'image traitée à sa taille d'origine
        Mat resizedEdgesRed = new Mat();
        Imgproc.resize(edgesRed, resizedEdgesRed, frame.size());

        // Superposer les contours rouges sur l'image d'origine
        Core.add(frame, resizedEdgesRed, this.processedImage);

        // Libérer les ressources inutilisées (libère explicitement la mémoire des objets temporaires)
        grayImage.release();
        flouImage.release();
        edges.release();
        dilatedEdges.release();
        edgesRed.release();
        resizedFrame.release();
        resizedEdgesRed.release();

        detection = false;
    }
    new tempo(1);
}
```

Explication :

- ▶ Boucle de traitement :
- ▶ Réduire la taille de l'image pour un gain de performance
- ▶ Transformé l'image en une nuance de gris
- ▶ Appliqué un filtre bilatéral
- ▶ Appliqué une dilatation pour mieux voir les contours
- ▶ Mettre les contours en rouge
- ▶ Réagrandir l'image traitée
- ▶ Superposer les contours rouges sur l'image d'origine
- ▶ libérer la mémoire

Résultat de la détection de contour sur la vidéo

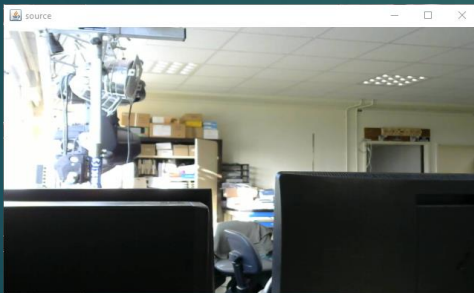
Image traiter :



Traitement du flux vidéo

► Détection de forme

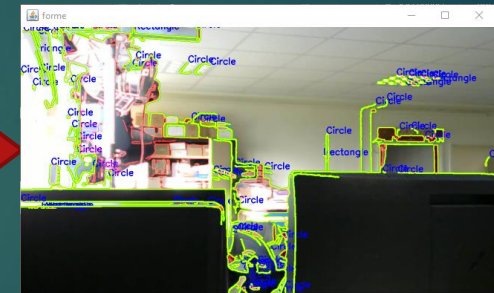
► Vidéo reçu :



► Traitement :

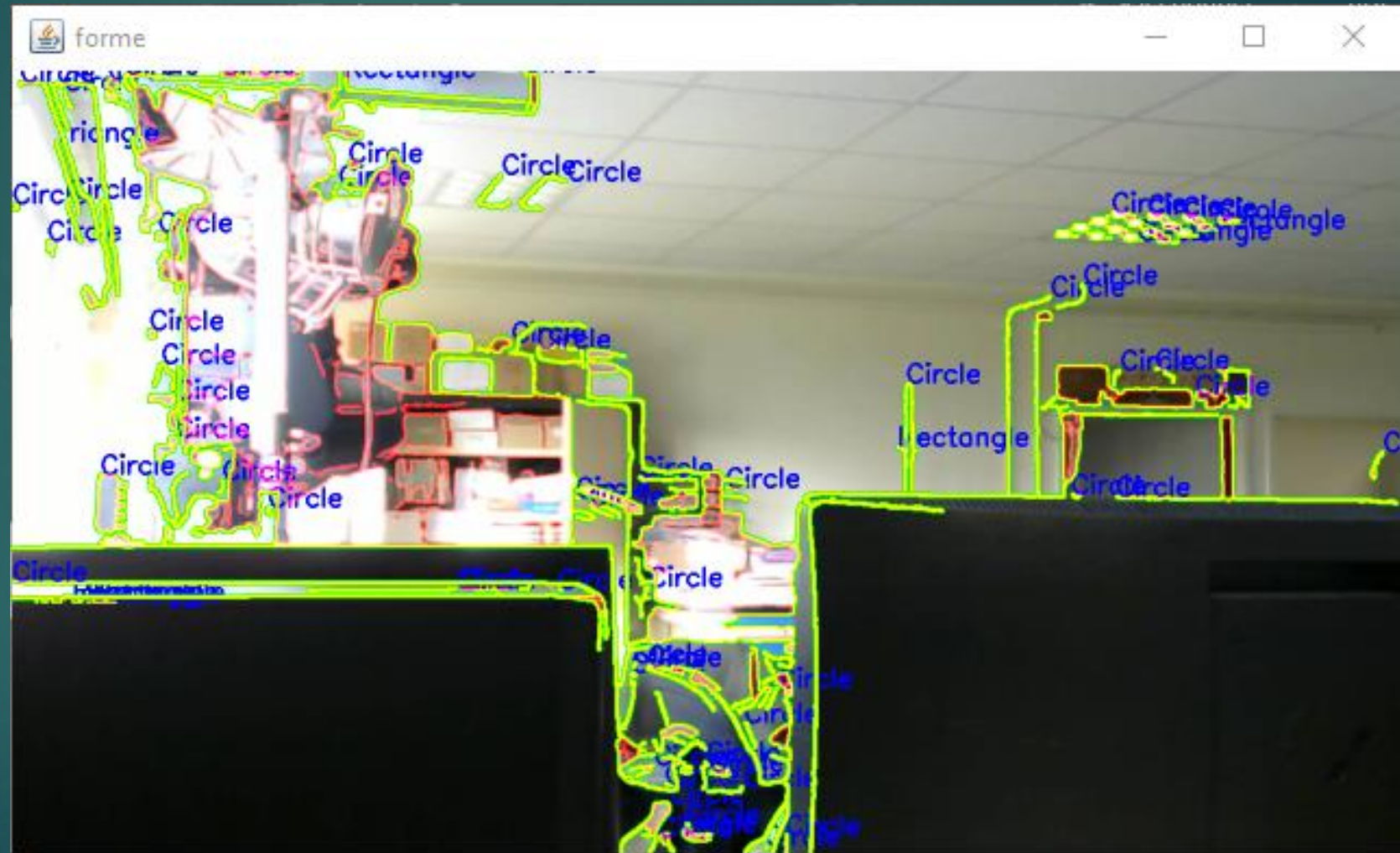


► Vidéo traiter :



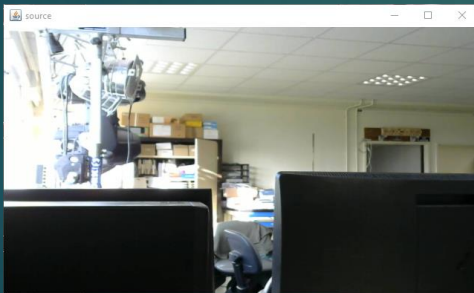
Résultat de la détection de forme sur la vidéo

Image traiter :



Élaborer un environnement 3d

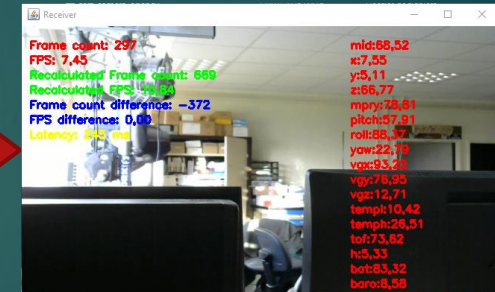
- ▶ Intégrer la vidéo
- ▶ Intégrer la télémétrie du char
- ▶ Vidéo reçu :



- ▶ Traitement :



- ▶ Vidéo traiter :



Résultat de l'affichage de la télémétrie dans environnement 3d

Image traiter :

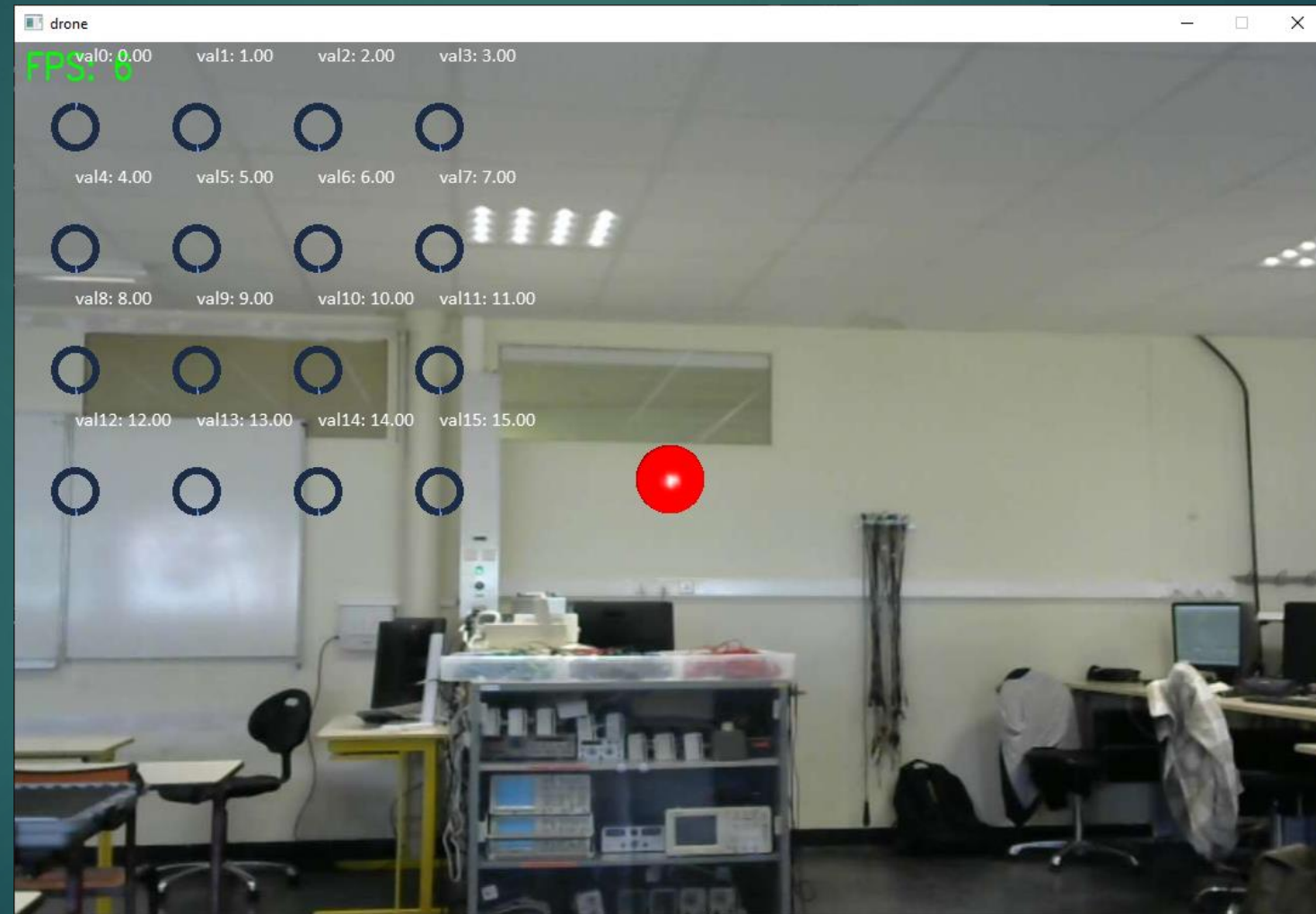
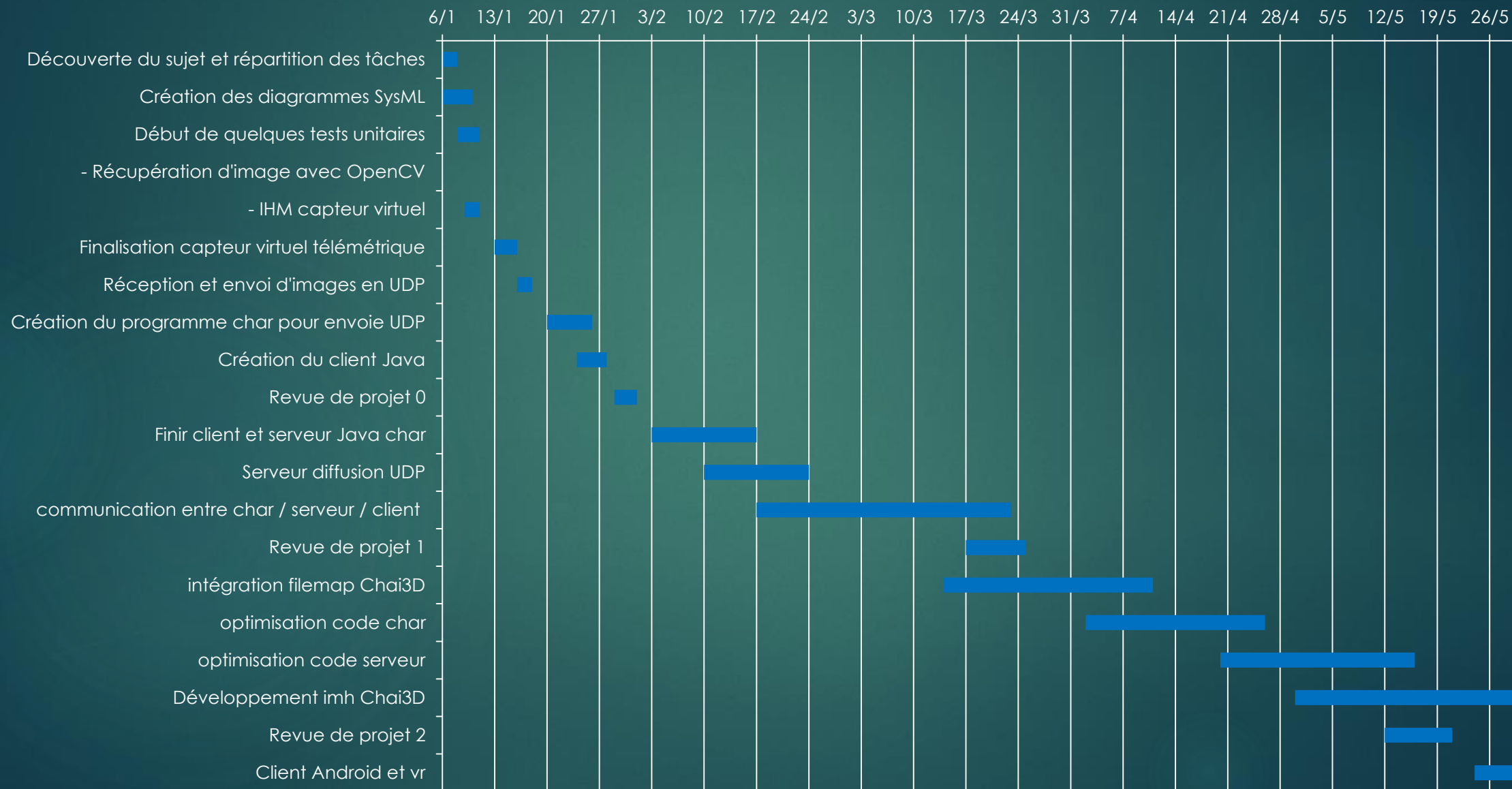


Diagramme de Gantt



Fin.