

PROJET INFORMATIQUE : Détection d’adventices à l’aide d’un drone



Bottais Théo / Le Troquer Célia

# PLAN :



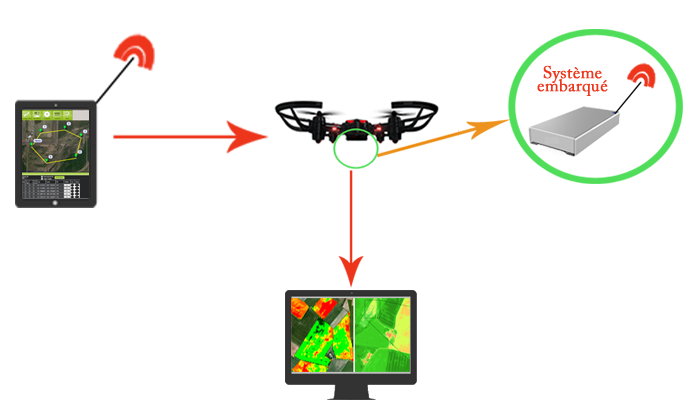
Le projet « Détection d’adventices à l’aide d’un drone », consistait à munir un drone en survol d’une zone de culture, d’un système permettant la prise de photos au-dessus de cette zone. Les photos devaient servir une fois le vol du drone terminé, à reconstituer une carte de la zone et à détecter après traitement d’image, les adventices dans la parcelle de terrain étudiée.

Dans le temps qui nous était impartit, nous ne pouvions traiter le projet dans sa globalité, nous avons donc fait le choix d’en traiter une partie qui est composée :

* Du développement du système embarqué à positionner sur le drone pendant son vol, chargé de la prise d’informations sur la zone étudiée.
* De la synchronisation du système embarqué avec le logiciel de planification de vol du drone.
* De l’exploitation ultérieure des captures photos afin de reconstituer une carte de préconisations.

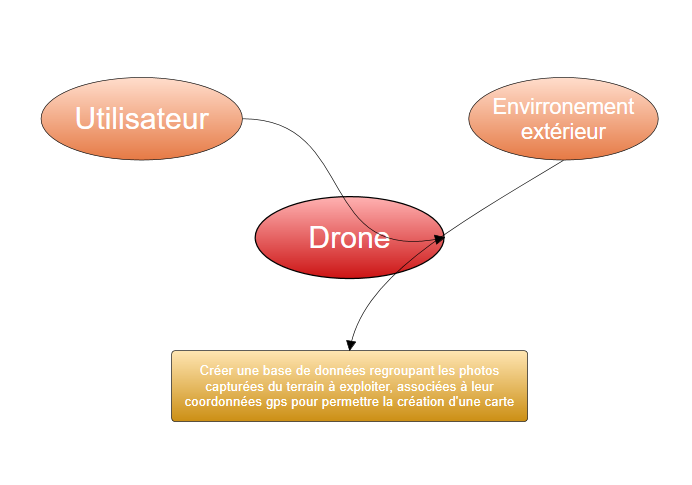
Nous n’avons donc pas pu traiter la partie traitement d’image qui consistait en la détection sur des photos en fausses couleurs, des zones comportant les adventices.

Nous allons dans cette première partie, présenter le cahier des charges que nous avons à partir de cela, développé et qui nous a permis de définir les besoins précis de l’utilisateur ainsi que les limites qui s’imposaient à nous.





Le schéma bête à corne suivant met en évidence les besoins de l’utilisateur, auxquels notre système se devait de répondre :



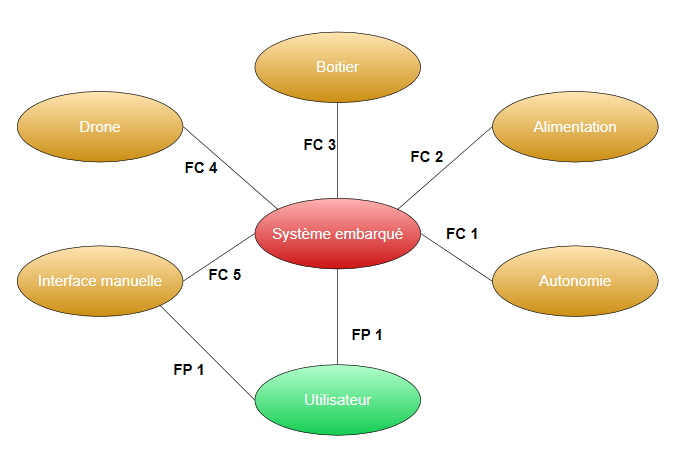
= Système embarqué

Avant toute recherche sur le choix des solutions techniques possibles, il nous a fallu fixer les caractéristiques et les différentes contraintes auxquelles notre système se devait de répondre. Nous avons ainsi pu élaborer le cahier des charges.

Nous en avons retiré que notre système se devait de :

* Etre composé d’un système de vol de type drone afin de prendre des captures du terrain en hauteur afin de détecter une large zone, typique des cultures importantes.
* Posséder un système embarqué autonome capable d’effectuer la tâche en complète autonomie, sans intervention humaine une fois lancé.
* Le drone doit posséder un logiciel de guidage automatique lui indiquant le trajet à parcourir.
* Le système embarqué doit être capable de capturer des photos, et donc être équipé d’une caméra en position de vue verticale, ce de manière répétée à un intervalle donné.
* Il doit également être capable de relever pour chaque photo capturée, les coordonnées GPS au même instant que la capture, afin de localiser chaque prise de photo, et donc doit être équipé d’un module GPS.
* Le système se doit donc également de posséder un système de stockage afin de créer un répertoire contenant toutes les captures.
* Le système doit être capable d’associer chaque photo à sa/ses coordonnée(s) GPS avant de l’ajouter au répertoire.
* Notre système entier doit pouvoir être commandé au départ pour être démarré en temps voulu et cette commande doit démarrer le logiciel de guidage ainsi que le programme de capture photos.
* Le système doit enfin posséder une source d’alimentation et un boitier à l’épreuve des conditions extérieures pour protéger les composants additionnels au drone.
* Ultérieurement, les photos devront pouvoir être assemblées pour former une seule et même image qui servira de carte.

Le diagramme pieuvre suivant représente les contraintes liées à l’élaboration de notre système :



**FP 1 :** Pouvoir lancer le système embarqué à l’aide d’une interface pour qu’une fois lancé, le système soit capable de parcourir le trajet demandé, et durant ce trajet, créer une base de données dans un répertoire, contenant : les captures photos prises, associées à leurs coordonnées GPS à un intervalle régulier.

**FC 1 :** Le système embarqué doit être capable de commencer son programme automatiquement au lancement de son parcours par l’utilisateur : à la simple sélection de la trajectoire à parcourir, par l’utilisateur, le drone doit partir et le programme du système embarqué doit s’effectuer à ce départ automatiquement sans aucune autre intervention de l’utilisateur.

**FC 2** : Le système doit posséder également une autonomie d’approvisionnement en énergie : il devra posséder une alimentation portable de type batterie.

**FC 3** : Le système tout entier rattaché au drone devra être contenu dans un boitier adapté aux capacités du drone et adapté aux conditions extérieures.

**FC 4** : Evidemment, pour la prise de photos exploitables, le système doit être placé à une altitude suffisante, ce qui peut être assuré par l’utilisation d’un drone.

**FC 5** : La présence d’une interface connectée au drone à son point de départ est nécessaire pour pouvoir sélectionner le trajet à parcourir et démarrer le programme.



Nous avons définis plusieurs critères de choix pour respecter les prérequis minimums du projet afin de répondre aux besoins mais également en tenant compte des contraintes que nous avons établies, qui sont les suivantes :

* Le coût des composants
* La compatibilité
* La taille et les conditions extérieures (pour le dispositif embarqué à bord du drone)
* L’efficacité du dispositif entier
* L’autonomie du système
* **Composition :**

Notre système doit assurer les fonctions principales suivantes :

|  |  |
| --- | --- |
| Relevé des données images et GPS | Construction d’une ortho mosaïque |
| Le système embarqué à bord du drone doit en effet être capable au décollage du drone, de capturer des photos de manière régulière et constante ainsi que de relever les coordonnées GPS au moment de la prise de la photo. Ces données doivent pouvoir être stockées sur une mémoire interne exploitable. | Une fois les données de la carte mémoire interne récupérées, elles doivent pouvoir être exploitées par un programme permettant de reformer la carte dans son ensemble afin de former après traitement d’image, une carte de consignes de traitement. Cette carte formée est appelée ortho mosaïque. |

* **Composants choisis :**

La sélection des composants concernait principalement la partie acquisition de données et stockage de celles-ci puisque l’exploitation se fait elle, par le biais d’un seul programme informatique, qui ne nécessite aucun matériel particulier, si ce n’est un ordinateur comme interface (pour lancer le programme).

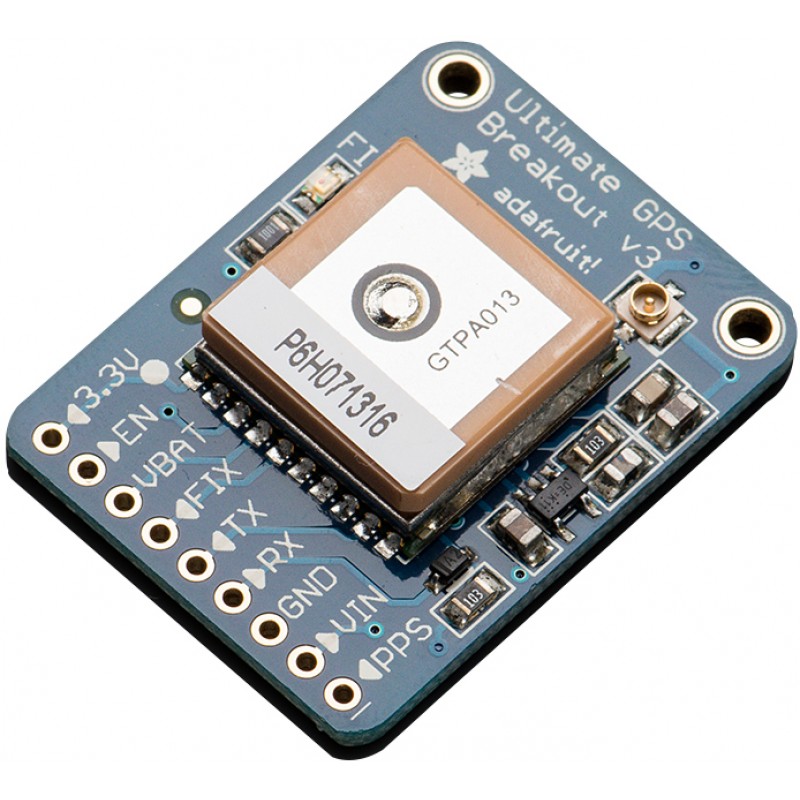
Nous nous concentrons donc dans cette partie, sur les composants que nous avons choisis pour constituer le système embarqué à bord du drone.

Dans un premier temps nous avons choisi, en fonction des critères cités précédemment, un nano- ordinateur capable de gérer les fonctions voulues à bord du drone. Pour cela nous avons travaillé sur une Raspberry Pi M.1 sous Linux.

Il nous a fallu par la suite penser à ajouter un module camera pour la capture photo. Pour cela nous avons utilisé un module Raspberry incluant directement le module camera pour la raspberry modèle V2 :



Enfin, notre système devant relever les coordonnées gps pour chacune des images afin de les situer les unes par rapport aux autres, nous avons dû choisir un module GPS à ajouter à la carte Raspberry. Nous avons donc choisi le module Adafruit ultimate breakout (datasheet en annexe dans le dossier documents) :



Nous l’avons ajouté au système grâce à un adaptateur USB/TTL qui nous a permis de brancher directement le module en USB au boitier Raspberry + caméra.

Aussi nous avons pris en compte la nécessité d’ajouter au système un stockage de masse USB à la raspberry pour récupérer les informations, ainsi qu’une batterie externe pour rendre le système autonome. Enfin nous avons eu à notre disposition un drone prêté par l’école : le bebop Parrot.



* **Automatisme du système :**

Pour la partie acquisition, il nous a fallu développer un programme capable de tourner dès le démarrage du système et donc dès le décollage du drone. Le programme se devait également de tourner en boucle afin de prendre des photos de manière régulière et continue. Pour cela nous avons appelé notre programme principal dans un fichier source du compte administrateur de la Raspberry. Ce fichier, appelé ***rc.local***, est exécuté à chaque démarrage de la Raspberry il nous a donc suffit d’appeler notre programme grâce à la ligne suivante, que nous avons directement insérée dans ***rc.local*** (voir code shell rc.local dans le dossier « Source ») :

***One.sh*** étant un fichier shell et notre programme principal.

Cette ligne désigne la destination qui est l’emplacement de notre programme.

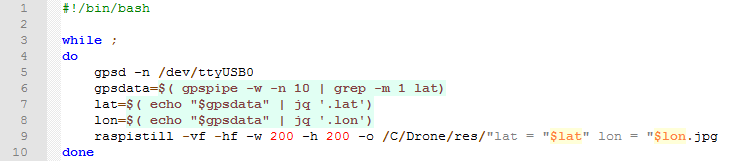
* **Programme d’exécution principal :**

La fonction principale du programme est de prendre des photos à un intervalle régulier grâce au module caméra intégré à la Raspberry, ainsi que de relever les coordonnées au même instant de la prise de la photo grâce au module GPS. La photo et ses coordonnées GPS doivent ensuite être enregistrées dans un répertoire sur un emplacement mémoire externe à la Raspberry : ici une clé USB. Nous avons donc pour cela, utilisée la bibliothèque shell du module caméra, afin de réaliser la capture d’une photo, la redimensionner et l’enregistrer, dans un premier temps :

Il nous a ensuite fallut relever les données GPS qui nous intéressaient, c’est-à-dire les données de latitude et de longitude. Pour cela nous avons sélectionné les dix premières données GPS reçu pour restreindre le nombre d’infos, puis nous avons extraites les deux données lon et lat grâce à une bibliothèque raspberry permettant la sélection de données dans un fichier texte de type JSON qui est le type de fichier retourné par le module GPS lord de l’acquisition des données :

Nous avons ensuite crée deux variables contenant pour une la latitude et pour l’autre la longitude :

Enfin nous avons choisi d’inclure ces données au nom donné à la photo lors de son enregistrement afin de permettre à l’utilisateur d’avoir tutes les données réunies :



*Fichier shell principal* ***one.sh***

Les images sont toutes enregistrées dans un fichier **res** qui est la base de données. Cette base de données est inclue dans le dossier « Drone » du projet java situé sur le stockage périphérique de masse USB appelé « C ». Ce projet java constitue la deuxième partie de notre système. Ainsi les données sont directement enregistrées dans un dossier sur la clé USB où se situe le programme java. Ce dossier étant lui-même partie intégrante de l’arborescence du programme java, l’utilisateur n’a plus qu’à récupérer l’espace de stockage après l’atterrissage du drone, pour exploiter ces données. L’image est également redimensionnée à 200px x 200px pour permettre son exploitation, grâce à la commande «  –w 200 –h 200 » dans la dernière ligne de code qui redimensionne l’image avant de la renommer et de l’enregistrer.

Enfin, afin de faire tourner notre programme en boucle nous avons utilisé une simple boucle WHILE dans laquelle nous avons placé notre code.

Le laps de temps observé entre deux captures photos avec notre système, est de 6 secondes en moyenne, cela est dû au temps nécessaire aux opérations effectuées sur chaque photo ainsi qu’au temps nécessaire à leur enregistrement.

Partie THEO

THEO MONTAGE : raspberry + module camera + usb mémoire + drone + batterie + téléphone commande vol



|  |  |  |
| --- | --- | --- |
| Composant | Quantité | Prix unitaire |
| Raspberry Pi | 1 | ~40 € |
| Module camera V2 | 1 | 30 € |
| Module GPS Adafruit | 1 | ~37 € |
| Adaptateur USB/TTL | 1 | ~20 € |
| Clé USB de stockage | 1 | ~5 € |
| (Drone Bebop) ? | 1 | ~450€ |
| Batterie externe | 1 | ~90 € |
| TOTAL | 5 | ~672 € |

THEO (si tu as le courage, sinon supprime ^^)



THEO :

Ouverture sur la nécessité d’un filtre ou d’une caméra NDVI pour obtenir images fausses couleurs, et sur le traitement d’image de la carte obtenue en fausse couleurs à l’aide d’OpenCV pour entourer les zones d’adventices + critique du système raspberry un peu trop lent entre deux captures.