

Projet Njord

Topographie d'une zone par communication avec une équipe de drones

Aigreault Clément, Henrio Jordan, Pham Chitin

16 Mars 2015

- Environnements inaccessibles par l'être humain
- Mission d'urgence, chantier...
- Besoin d'un intermédiaire

- Technologie à un stade intéressant
 - Robotique
 - Intelligence artificielle
 - Communication sans fil
- Possibilité de "donner vie" à des machines
- Une perte matérielle est moins importante qu'une perte humaine

- Élaboration d'un réseau étoilé
- Le noyau : un serveur
- Les branches : drones volants
- Les drones récoltent des informations
- Le serveur cartographie la zone

Introduction

Solution

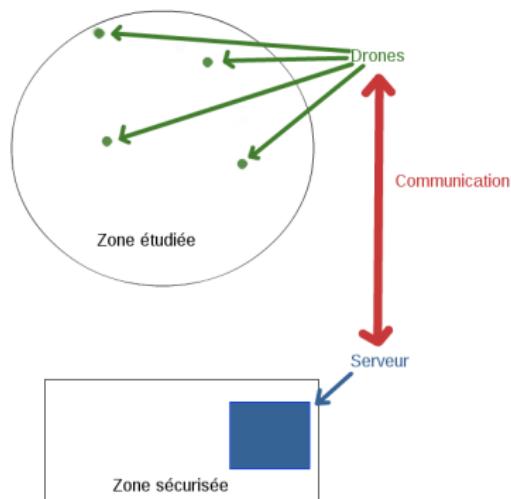


Figure : Schéma du réseau

- Drones autonomes et équipés d'un capteur ultrason
- Communication par fréquences radio
- Représentation graphique de la zone en temps réel

Déroulement de la présentation

- 1 Serveur
- 2 Drone
- 3 Analyse
- 4 Conclusion

1 Serveur

- Principe
- Développement
- Démonstration

2 Drone

3 Analyse

4 Conclusion

- Traite les informations recoltées
- Chaque drone connaît le serveur, mais pas les autres drones
- Le serveur connaît tous les drones
- Possibilité d'envoyer des ordres

- Constitué de trois entités distinctes
 - Communication
 - Sauvegarde des données
 - Cartographie
- Concurrence
- Limiter la perte d'informations

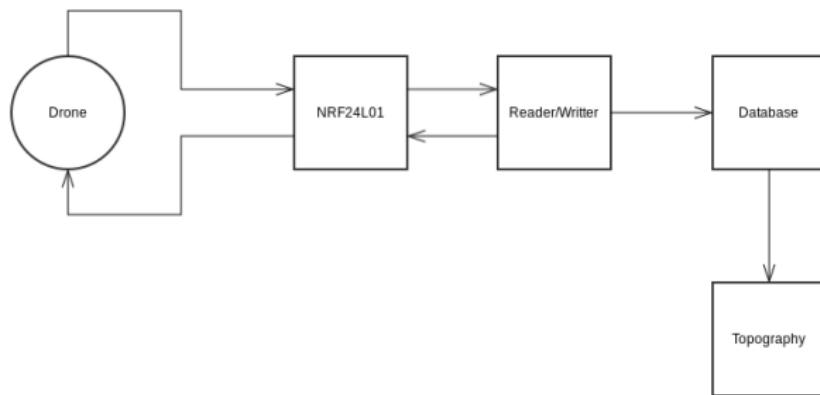


Figure : Modèle du serveur

Tâche 1 : Communication

- Lit les messages des drones
- Écrit sur le port série de la machine
- Lit le port série
- Envoie des ordres aux drones

Tâche 2 : Sauvegarde des données

- Lit le port série
- Insère les messages dans la base de données (BDD)
- Écrit sur le port série

Tâche 3 : Cartographie

- Lit le contenu de la BDD
- Insère chaque entrée dans une matrice
- Dessine le contenu de la matrice
- Détermine si un ordre doit être envoyé

Tâche 1 : Communication

- Utilisation d'un NRF24L01 (composant)
- Implémentation en Arduino

Tâche 2 : Sauvegarde des données

- Implémentation en Python
- BDD implémentée à l'aide de Redis
 - Utilisation simple (fonctionnement, Python)
 - Système robuste comme SQL inutile

Tâche 3 : Cartographie

- Implémentation en Python
- Utilisation de Numpy et Matplotlib
- Matrice souvent redimensionnée

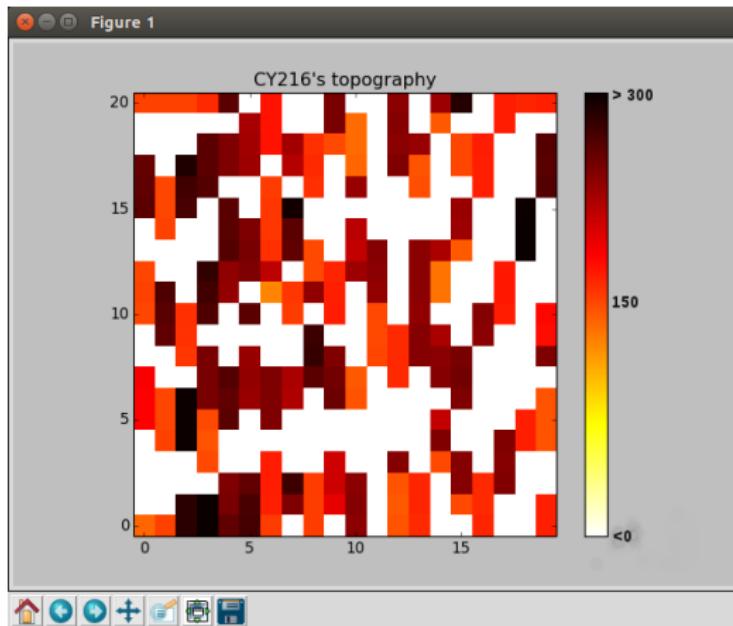


Figure : Exemple de rendu de topographie

Démonstration

1 Serveur

2 Drone

- Étude préliminaire
- Composants
- Développement et aquisition des pièces manquantes
- Résultat final

3 Analyse

4 Conclusion

- Réalisation du cahier des charges
- État de l'art
 - Crazyflie (BitCraze)
 - ARDrone (Parrot)
 - Phantom (DJI)
- Plusieurs type de drones (grand/petit, rapide/lent, agile/prise de vues...)
- Étape non négligeable d'un projet

- Coup de coeur pour le Crazyflie de BitCraze
- Pièces détachées, code source disponible, libre
- Un peu cher et ne correspond pas parfaitement à notre application



Figure : Photo du Crazyflie de BitCraze

- Devis
- Premières estimations
 - Poids
 - Puissance requise
 - Consommation en courant
 - Autonomie
- Plus lourd que le Crazyflie mais moins cher

Arduino

- Cours d'Arduino en début d'année
- Micro-contrôleurs très en vogue
- Similaire au C/C++
- Bon marché et existe en petite taille

Arduino

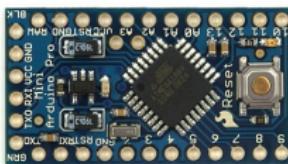


Figure : Photo de l'Arduino Pro Mini

Communication sans fil

- Nombreux composants sur le marché
 - Modules XBee : Très utilisé mais trop coûteux ($\sim 30 \text{ €}$) et trop gros
 - Fréquence radio : Assez utilisé, bon marché ($\sim 0,80 \text{ €}$) et petit
- NRF24L01 suffisant au sein d'une même pièce

Communication sans fil

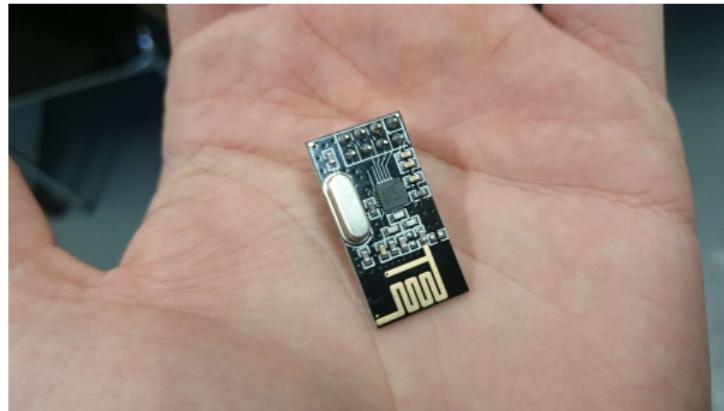


Figure : Photo du NRF24L01

Gyroscope/Accéléromètre

- Besoin de stabiliser le drone
- MPU-6050 très répandu, petit, bon marché et embarque un accéléromètre
- Besoin de déterminer la position du drone dans l'espace
 - GPS : Trop peu précis pour une salle
 - Triangulation : Trop coûteux, peu adapté à la problématique
 - Mesure de l'accélération
- Solution non exploitable

Gyroscope/Accéléromètre

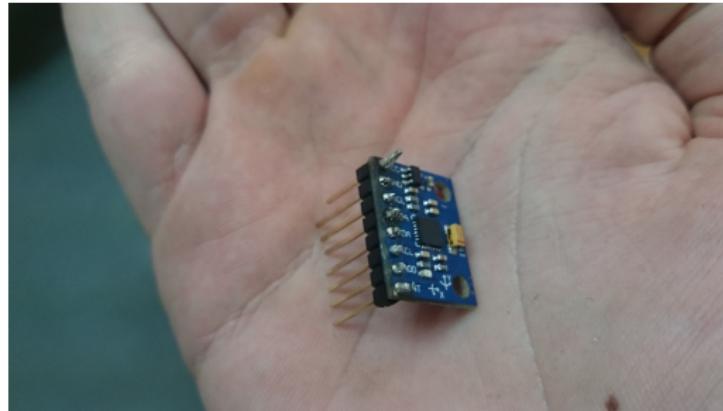


Figure : Photo du MPU-6050

ESC

- Electric Speed Controller(ESC)
- Coûteux ($\sim 15 \text{ €/ moteur}$)
- Lourd ($\sim 25\text{g} / \text{ESC}$)
- Fabrication possible (transistor)

Moteur, batterie, total

- Même moteurs et batterie que le Crazyflie
- ~ 25 € contre 120 €
- 35g contre 19g (estimation)

- Développement de bibliothèques composant avant assemblage
 - Moteur
 - Communicateur radio : Surcouche de RadioHead
 - Gyroscope : Inspiration de Jeff Rowberg

- Encore deux pièces manquantes
 - Circuit imprimé
 - Fixations moteurs
- Pas de matériel à l'EISTI, besoin d'aide extérieur
- Visite au fablab de Gennevilliers

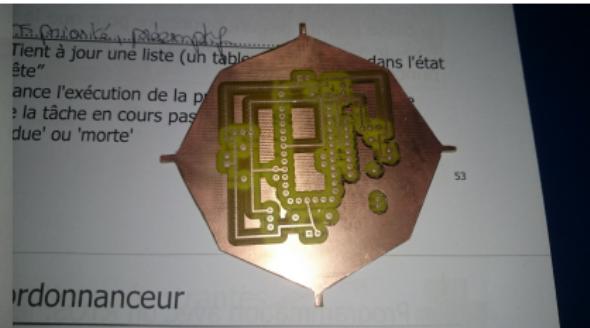
Circuit imprimé

- Dessin du circuit à l'aide du logiciel Fritzing
- Support de la part de l'ENSEA

Circuit imprimé



(a) Avant



(b) Après

Figure : Circuit avant et après découpage

Fixations moteur

- Modèle des pièces disponible sur GitHub
- Demande auprès d'une entreprise, mais pièces trop minutieuses
- Demande auprès de Polytech Instrumentation (filiale de Jeulin) lors de Robafis

Fixations moteur



Figure : Photo des fixations moteur

Après assemblage

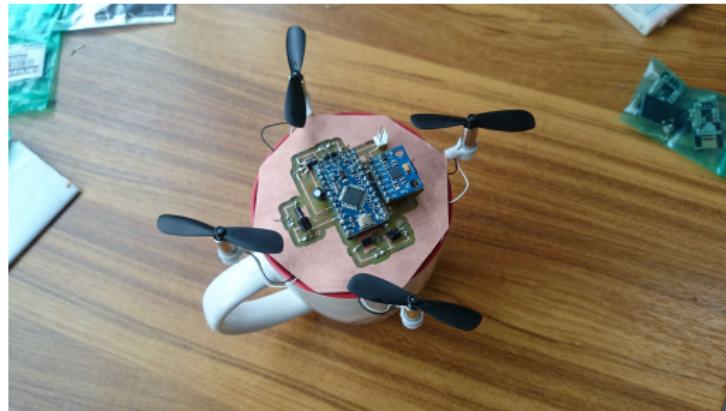


Figure : Photo du drone après assemblage

- Premier réflexe : Vérifier le décollage
- Plusieurs problèmes
 - Contrôle moteur depuis l'Arduino : Non suffisant, capacité maximum des moteurs non atteinte
 - By-pass de l'Arduino : Peut-être suffisant, un moteur défectueux

Démonstration

1 Serveur

2 Drone

3 Analyse

- Conception
- Contacts extérieurs
- Montage
- Expérience

4 Conclusion

- Erreurs dès le devis
- Estimation du poids (approximations et éléments négligés)
- Circuit imprimé élément le plus lourd
- 10% de la somme totale des composants électroniques

- Choix des moteurs
- Rapides mais couple insuffisant
- Ne peuvent pas supporter le poids du drone

- Moteurs : alimentés en 3,3V
- Capteurs ultrasons : nécessitent du 5V
- Drone non fonctionnel dès la conception

- L'EISTI peu adaptée à ce genre de projet
- Besoin de support extérieur (fixations moteur, circuit imprimé, découpe)
- Temps de recherche + création => Délai conséquent
- Premiers essais trop proches de la deadline (fin février)
- Pas le temps de profiter de l'expérience acquise pour une seconde version
- Date de tests idéale : Fin 2014

- Source d'achat des pièces
- Coût réduit se ressent sur la qualité (contrefaçon)
- Moteur provenant du même fournisseur, mais puissance différente
- Arduino différente de l'officielle, prise de retard

- Faire du circuit, notre châssis
- Deux inconvénients majeurs
 - Fiabilité de la fixation
 - Moteurs non parfaitement verticaux

- Formation assez généraliste
- Nous ne sommes pas de vrais ingénieurs système
- Projet complexe et nécessite plus de temps (au moins deux ans)
- Contraintes auxquelles nous ne sommes pas habitués (matériel, délai de livraison, système)

- 1 Serveur
- 2 Drone
- 3 Analyse
- 4 Conclusion

Implémentation

- Création d'une organisation sur GitHub (<https://github.com/NjordProject>)
- Découpage de l'ensemble du projet en plusieurs sous-répertoires
- Hébergement d'un blog (<http://njordproject.github.io/>)
 - français, anglais, japonais
 - Suivi du projet
 - Ressources pour d'autres projets
- Utilisation de technologies différentes
 - Arduino (hardware, software)
 - Python (Numpy, Matplotlib)
 - Redis

Serveur

- Quasiment fonctionnel
- Nécessité d'un drone fonctionnel

Construction d'un drone

- Challenge énorme
- Connaissances pluridisciplinaires et complexes
- Petites erreurs trop nombreuses
- Manque de temps pour profiter de l'expérience aquise

Détermination de la position

- Grande étendue en extérieur : GPS
- Pièce : Peu de solution
- Problématique au stade de la recherche

Expérience aquise

- Enrichissant techniquement parlant
- Mais aussi du point de vue relationnel
- Support de la part de contacts extérieurs
 - Nga Nguyen : Participation de Polytech Instrumentation
 - Besma Zeddini : Participation de l'ENSEA
- Merci !

Merci pour votre attention