Rapport de Stage fin de DUT - SuperBeeLive

Olivia SERENELLI-PESIN

July 24, 2019

Remerciements

J'adresse mes remerciements aux personnes qui m'ont permis de réaliser ce stage dans l'équipe de SuperBeeLive.
Tout d'abord Matthieu ROUSSET, initiateur du projet SuperBeeLive à l'IBMM (Insitut Biomoléculaire Max Mousseron) qui m'a accueilli au sein de son équipe.
Ensuite, Capucine CARLIER pour ses nombreuses explications biologiques sur les abeilles ainsi que sa disponibilité afin de comprendre au mieux les enjeux et les besoins des biologistes pour mon projet.
Enfin, Sebastien DRUON pour m'avoir encadré, aidé à m'intégrer dans le milieu de la recherche et aidé sur une multitude de sujets, aussi bien du point de vu universitaire que sur les tâches qui m'ont été confiées.

Contents

Chapter 1

Introduction : Présentation du projet de recherche

1.1 La structure d'accueil

Le projet étant réalisé par plusieurs laboratoires de recherches, j'ai dû évoluer au seins de différentes organisations. Tout d'abord il y a mon employeur, l'Université de Montpellier, qui englobe d'autres structures où j'ai pu évoluer. L'université a été, pour moi, une entitée administrative.
Avec elle, le CNRS (Centre National de Recherche Scientifique) accueille dans ses locaux le rucher expérimental où nous pouvons effectuer nos tests d'installation pour le projet. J'ai pu m'y rendre plusieurs fois afin d'observer les abeilles et voir le travail déjà effectué et évoluer au fur et à mesure. C'est également chez eux que nous aurons un serveur d'installé.
Ensuite, l'IBMM est le laboratoire qui a engagé l'argent lié au projet afin de pouvoir me recrutrer lors de ce stage. C'est également une structure qui a été seulement administrative de mon point de vu puisqu'ils m'ont envoyés travailler dans les bureaux du LIRMM (Laboratoire d'informatique, de robotique et de microéléctronique de Montpellier), autre laboratoire de recherche, afin que je puisse être aux côtés de Sébastien DRUON qui m'aura donné la grande majorité de mes tâches à effectuer. J'ai pu y avoir mon bureau en face de M. DRUON, me permettant d'avoir une certaine autonomie mais aussi de pouvoir faire appel à lui facilement lorsque j'en avais besoin.

C'est dans ce contexte de recherche que j'ai pu découvrir et travailler sur le projet SuperBeeLive.

1.2 Projet SuperBeeLive

1.2.1 Le Global

La santé et le développement des abeilles est aujourd'hui une question de plus en plus étudiée. Les bouleversements majeurs de notre planète et de l'activité humaine se traduisant par une augmentation alarmante de la mortalité des colonies et une chute de la production du miel dans nos pays développés, il est urgent de se préoccuper de leur futur. La situation des abeilles domestiques alerte le pouvoir public sur l'accélération de la dégradation de la biodiversité des pollinisateurs domestiques et sauvages, et de la flore qui en dépend. Ces dégâts sont dûs, entre autre, à l'apparition et la prolifération d'espèces invasiges pour les abeilles, provoquant maladies et déteriorations.

Le projet consiste en la structuration de plusieurs collaborations existantes ou nouvelles autours du developpement d'une ruche plate instrumentée destinée au monitorage détaillé de la santé de l'abeille et des écosystèmes. Son but est de pouvoir répondre à des questions clés, notamment autours des mécanismes physiopathologiques et des maladies chroniques dûes au parasites ainsi qu'aux altérations de l'écosystèmes et des qualités nutritives des produits des ruches.
Répondre à ces questions permettra de regrouper différentes solutions technologiques systèmatiques, automatiques et non invasives à la collection de données usuelles déterminantes dans chacun des thèmes abordés. Les différentes travaux déjà effectués autours de ce sujet ne visaient qu'un seul type de problème à la fois, ne permettant pas une vision globale des difficultées rencontrées par les abeilles. Notre but est de réunir les différentes données qui peuvent être utilisés pour étudier l'influence des éléments et événements extérieurs sur leur santé et leur cadre de vie.

Concrétement, l'équipe de SuperBeeLive va concevoir une ruche plate afin d'y mettre en place plusieurs type de capteurs (hygrométrie, vibrations, température interne et externe, etc) ainsi que des caméras qui filmeront l'intérieur et l'extérieur de la ruche.

Cette instrumentation nous permettra dans un premier temps d'observer les abeilles afin de visualiser leurs comportements (danse, regroupement, protection, etc) ainsi que les parasites comme les frelons asiatiques ou les varroas. Une fois ces observations faites et documentées, il y aura assez de matière pour pouvoir créer des algorithmes qui reconnaitront automatiquement ces comportements pour en sortir des informations spécifiques.
Par exemple, un des buts et de repérer la danse d'une abeille qui permet aux autre d'indiquer où se trouve du polen et d'en extraire les informations qu'elle transmet (la direction à prendre, le temps à parcourir pour y arriver etc).
Avec ces recherches, il serait possible de mettre en place une vitrine web des caméras streamées et commentées automatiquement en temps réel, permettant à d'autres chercheurs d'avoir une ressource permanente pour travailler mais aussi au grand public d'avoir un accès plus restreint mais pédagogique sur le comportement des abeilles.

1.2.2 Mon rôle durant le stage

Pour ce stage, plusieurs missions m'ont été affectées, une qui est devenue ma principales et d'autres, plus courtes dans le temps.

Comme dit plus haut, le but premier de l'installation des caméras sur la ruche est de pouvoir observer et annoter des vidéos manuellement pour ensuite rendre cette tâche automatique. Seulement, ces annotations doivent être encadrées afin d'éviter que des données ne se perdent, et que les outils utilisés pour le faire ne soient pas les même d'une personne à une autre nous donnant alors des fichiers non uniformes et plus difficiles et long à traiter une fois réunis.
Ainsi, nous avons entreprit de créer un logiciel d'annotation, permettant de regarder en direct les caméras et de sauvegarder des morceaux de vidéos afin de pouvoir dessiner simplement dessus (encercler, mettre une flèche, encadrer, etc) et écrire quelques mots sur ce qu'on y observe. Ces vidéos seraient sauvegardées dans un fichier contenant la vidéo et les annotations et pourront être visualisés de nouveau dans ledit logiciel mais aussi dans un lecteur plus classique mais sans les annotations.
Au final, celui-ci alégera le travail des biologiste, qui auront un outils sur mesure pour annoter les vidéos, mais aussi de M. Druon, pour qui il sera plus simple de récupérer et traiter les données.

Il est évident que dans un tel projet, beaucoup de données seront transmises et stockées. Ainsi, toute une partie d'administration système est à gérer. Dans notre cas, nous avons deux tâches importantes.
D'abord, la question du stockage des données commençait à se poser lors de mon arrivée en stage. Il fallait choisir, acheter, installer puis configurer un serveur de stockage dans la salle serveur du CNRS.
Ensuite, le rucher du CNRS où notre ruche expérimentale est installé n'a pas de configuration réseau déjà établie. Une connexion par fibre optique est prévue, mais la suite de l'installation devra être gérée par nous même. Comme pour le serveur, il faudra choisir, installer et configurer un switch dans le rucher.
Pour ces deux tâches, la même problématique est soulevée : il faudra penser au grand nombre de données qui devront transiter sur le réseau et donc prévoir du matériel adapté.
D'autres réalisations auraient pû m'être affectées, comme le dimensionnement des caméras ou la construction des cartes éléctroniques qui seront installées au centre de la ruche. Seulement, ces sujets s'éloignant de mon DUT Réseaux et Télécommunications et mes compétences et connaissances dans ces deux domaines étant limitées, je n'ai pas eu à travailler dessus.
Cependant, la possibilité d'un apprentissage lors d'école d'ingénieurs en systèmes embarqués a été évoquée, ce qui correspondrait plus à ces tâches.

1.3 L'équipe

Brouillon organigrame

CNRS : Jean-Baptiste THIBAUD 50% IBMM : Matthieu ROUSSET 50% LMGC : Delphine JULIEN Capucine CARLIER LIRMM : Jean TRIBOULET Sébastien DRUON IUT de Béziers : Philiphe PUJAS -¿ stockage des données Equipe Bee@Wur Université Wegeningen (Néerlandais) -¿ ouverture internationnalle.

Chapter 2

Projet principal

2.1 Analyse du besoin et des fonctionnalités exigées

Dans le cadre de la récolte et de l'analyse des données, il était primmordial pour l'équipe d'avoir l'outils d'annotation décrit plus haut.
Celui-ci devait permettre de :

- Visualiser les caméras en direct
- Démarrer la capture vidéo à tout moment
- Ajouter un système de tag par mots clés afin de pouvoir trier facilement les vidéos
- Avoir plusieurs types d'annotations (entourer, marquer, avoir des mouvements etc)
- Retrouver toutes les mesures de la ruches (température, horaires, numéro de caméra, de ruche, de cadre etc)
- Avoir un principe d'auteur
- Pouvoir revisualiser les vidéos
- Pouvoir remodifier les vidéos

Afin de répondre à ces besoins, nous avons imaginé l'interface suivante : Bien sûr, celle-ci a connu beaucoup d'évolutions au cours de son développement : au fur et à mesure de l'avancement, nous nous rendions compte de certaines éventualités que nous n'avions pas imaginé et que nous avons intégré à la volée.

2.2 Contraintes

Le langage de départ pour coder l'interface et ses fonctionnalités, que nous pourrons familiairement nommer partie moteur et partie physique, m'a été imposé.
C'est donc en C que j'ai dû réflechir à comment préparer et lier ces deux parties. Ce langage a été choisit tout simplement parceque c'est le langage que M. DRUON a pour habitute d'utiliser.
La partie moteur peut être développée en C sans trop d'ajout de librairies annexes autres que celles dites basiques (stdlib, stdio). Cependant, nous avons dû utiliser une librairie pour développer la partie physique. Nous avions le choix entre GTK ou QT. QT devant être utilisé en C++, notre choix s'est naturellement dirigé vers GTK 3.0.
Comme dans tout projets collaboratifs, la convergence des données est importante et peut parfois se réveler difficile à mettre en place. Heureusement pour nous, en programmation l'outil GIT est excellent pour travailler à plusieurs. L'ayant déjà vu en cours cette année, j'ai pu l'utiliser concrétement lors de mon stage. Cependant, avant de commencer à utiliser concrètement l'outil, il a été préférable que je me remette à niveau sur celui-ci et ai donc entammé la lecture du livre Mastering Git qui m'a été d'une grande aide pour avoir des bases solides. Pour héberger notre code, nous avons choisi de le déposer sur GitHub : il est donc accessible au public sous le nom superbeelive.

2.3 Création de l'interface

GTK est une boîte à outils constituée d'un ensemble de fonctions, permettant de réaliser des interfaces graphique. On le retrouve dans un bon nombre de logiciel connus : de base créé pour GIMP (d'où son nom, "Gimp Toolkit"), il est aussi présent avec LibreOffice, Eclipse, Thunderbird mais aussi sur les versions GNU/Linux de Firefox. Codé en C avec une programmation orientée objet, on peut utiliser GTK dans d'autres langages comme le C Java, Python, Perl, PHP... Une fois l'utilisation de GTK décidée, j'ai dû me former à l'utilisation de cette librairie en autodidacte et avec l'aide de mon seigneur et maître DRUON. Pour cela, le livre m'a servi de référence de base quant à la manière de manier les éléments. Je me suis également aidée de divers tutoriels sur internet, mais surtout de la document officielle de GTK.
Souvent, on a tendance à utiliser Glade, un logiciel permettant de créer une interface GTK avec des outils graphiques. Dans mon cas, je m'en suis surtout servie en tant que bibliothèque pour voir et tester certains Widgets ainsi que leur fonctionnement. Cet utilitaire, certes plus rapide à prendre en mains, est lié au fait qu'il utilise des fichiers XML pour décrire l'interface ce qui, au final, complique la portabilité de l'interface d'une version du logiciel à l'autre. De plus, j'ai préféré apprendre à utiliser GTK en ligne de code "brut". Plus fastidieu au départ, j'ai trouvé cela plus simple à la longue : je maîtrisais vraiment mes outils et apprenais plus rapidement à utiliser un nouvel élément de la librairie.

2.3.1 Principe de GTK

Apprendre à utiliser GTK a été, pour moi, un peu long au début et m'a nécessité beaucoup de temps passé à poser calmement et méthodiquement les choses afin de comprendre sa logique. Si on a déjà fait de la programmation objet, il est évident que l'apprentissage de cette librairie n'en sera que plus simple.
L'idée principale de GTK est que l'on va ranger tous les éléments en fonction d'un autre élément, pour finir avec des sortes de poupées russes qui s'emboîtent et se rangent côte à côte pour donner un résultat final. Il exite une multitude d'éléments utilisables, appelés les Widgets. L'élément de base widgets contient des propriétés de base et, pour créer d'autre type d'élément, les développeurs de GTK y ont ajoutés d'autres propriétés à celles déjà existantes. Au fur et à mesure, ces ajouts de propriétés ont permit de créer toute sorte de Widget : des textes, des labels, des tableaux, des séparateur, des listes, des bouttons... Ces héritage et cette hierarchie des objets va nous construire un arbre d'élements, nous permettant de savoir ce que l'on peut faire avec nos Widgets : En effet, si par exemple je veux changer un paramètre d'un élément "GtkSpinButton", et que je ne trouve pas de fonction affectant directement ce type d'élément, alors je vais regarder si chez ses parents je peux trouver le paramètre voulu, à savoir "GtkEntry", et "GtkWidget".

Chaque paramètre disponible pour les Widgets peut être changé directement à l'aide de fonctions retrouvable dans l'excelente documentation en ligne de GTK 3.0. Il est totalement déconseiller de changer directement la valeur des paramètre à la mains en utilisant, par exemple des pointeurs. Il faut partir du principe que chaque élément a son nombre de fonctions associées et que tout est prévu pour pouvoir faire ce que l'on veut sur notre interface.

2.3.2 Des boites, dans des boites...

Une fois la manière de créer les éléments comprise, il faut ensuite comprendre comment les ranger. D'abord, on va créer une fenêtre, soit une GtkWindows. Dans celle-ci, on ne peut y poser qu'un seul Widget. C'est pour ça que nous avons les éléments "GtkContainers" : c'est avec eux que nous allons pouvoir structurer et placer tous les autres éléments dans notre fenêtre. Le container de base est la box. A sa création, nous devons simplement indiquer si celle-ci va être horizontale ou verticale, c'est à dire si les éléments que nous allons mettre dedans vont être placé de haut en bas ou de droite à gauche. Une fois cette première GtkBox placée, le problème de limitation de place imposée par la GtkWindow n'en est plus un. Maintenant, nous pouvons ranger ce que nous voulons dans cette boite. C'est là que le système de poupée russe prend son sens : nous allons devoir jouer avec les différentes boîtes pour créer les espaces que nous désirons. Voici le schéma de construction que j'ai fais pour la première fenêtre de l'application : On peut voir que la GtkBox box_principale prend toute la place de

ma fenêtre GtkWindow. Dans cette box_principale, j'y ai rangé verticalement box_up et box_down. Dans box_down j'ai integré box_left et box_right horizontalement, me permettant d'avoir quatre zones délimitées. Ensuite, j'ai créé des box pour mes éléments plus précis : dans box left j'ai directement intégré box_video dans laquelle se trouve la vidéo. Dans box right on va retrouvé box_info, box_meta, box_btn_cam, box_btn_video, box_info_time et box_file. Chacune de ces box m'ont permit de ranger à l'intérieur les éléments que je voulais retrouver : les boutons, les textes, les descriptions etc. A noter qu'en plus de ces boîtes j'ai également parfois ranger des séparateur, les barre horizontale et verticale, permettant d'ajouter un peu de structure visuelement.
Une fois que l'on sait comment ranger les éléments sur une fenêtre et quel est le principe de ces éléments dans l'idée, il ne manque plus qu'à voir comment concrétement, dans le code, on applique ces principes.

2.3.3 Déclaration

2.3.4 Placement, cosmétique et affichage

2.3.5 Signaux et fonctions

2.4 Définition des fonctionnalités

2.4.1 Les structures

2.4.2 Les vidéos

Chapter 3

Acquis et compétences

3.1 Missions annexes

3.2 Compétences développées

Chapter 4

Conclusion

Chapter 5

Références

5.1 Annexes

5.2 Biblio

5.3 Lexique