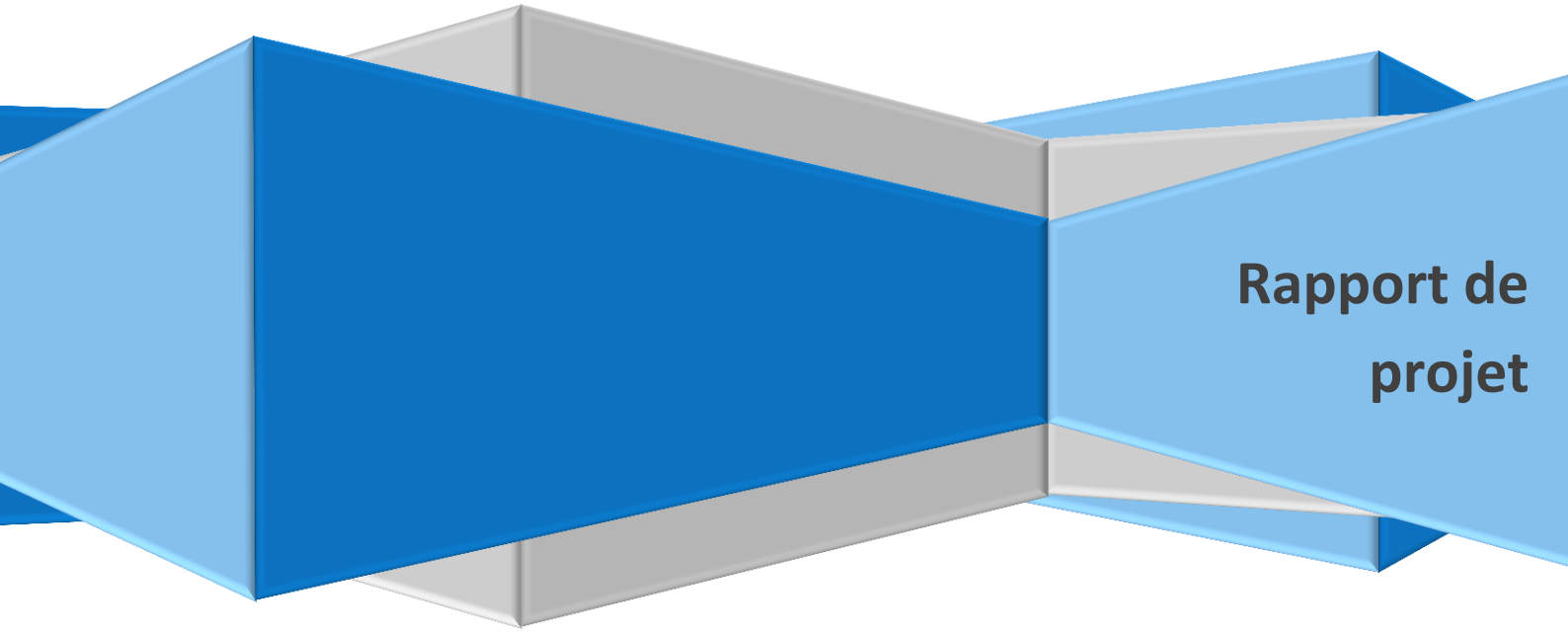
Université François Rabelais de Tours
Polytech Tours
Département informatique industrielle

Application de détection pour étiquette de bouteille de vin

Projet de fin d'étude



Rapport de projet

Emilie LEMEE

Encadrant :
Mathieu DELALANDRE

Sommaire

Introduction

Lors de la cinquième année d'ingénieur à Polytech Tours, nous devons mener à bien un projet de fin d'étude. Ce projet a pour but de mettre œuvre les connaissances acquises lors de la formation. Nous mettrons aussi un point important sur la gestion de projet au vu des documents à rendre tout au long du projet.

Le projet que j'ai choisi de réaliser est une application Android de scanner à vin. C'est une application qui permet d'obtenir les informations d'une bouteille de vin à partir de son étiquette. Le projet sur lequel je travaille est une partie de cette application de scanner à vin, elle se concentre sur la partie détection de l'étiquette.

J'ai choisi de mener à bien ce projet car, bien que n'ayant pas beaucoup de connaissances, les applications mobiles prennent une place de plus en plus importante dans notre vie quotidienne. Il me semblait donc important de développer mes compétences dans ce domaine.

Ce dossier vous présentera le travail effectué au cours du projet, nous verrons d'abord la présentation du sujet, son contexte et la problématique. Nous aborderons ensuite la partie « cahier des charges » dans laquelle je présenterai les choix qui sont mis en œuvre. Nous verrons ensuite la partie analyse et enfin la partie réalisation.

Scanner à vin et contexte du projet

Dans cette partie, je vais vous présenter le projet dans sa globalité. Nous aborderons la notion de scanner à vin et nous verrons les applications qui sont existantes aujourd'hui. Nous en tirerons la problématique et nous verrons plus précisément la partie de mon projet : la détection d'étiquette.

Scanner à vin

Un scanner à vin est une application pour terminal mobile (smartphone ou tablette) qui a pour but d'obtenir des informations sur une bouteille de vin.

Les bouteilles de vins possèdent une esthétique travaillée qui est généralement très importante pour le marketing. Il est impossible d'ajouter un QR Code sur l'étiquette de la bouteille sans modifier et perturber l'esthétique de la bouteille de vin.

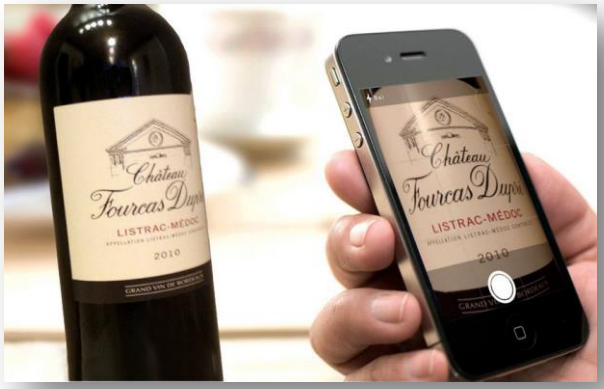


Image 1 : Illustration du scanner à vin

Identifier les bouteilles de vin se fait alors uniquement grâce à la reconnaissance d'étiquettes. C'est pourquoi des applications de scanner à vins sont créées.

Elles permettent de visualiser la bouteille via le smartphone, l'étiquette de la bouteille est alors détectée, elle est identifiée et les informations s'affichent sur l'écran de l'utilisateur. Il peut alors savoir de quelle année date la bouteille, avec quels plats le marier, ainsi que l'avis des utilisateurs.

Applications existantes

Il existe déjà des applications de scanner à vins. Les applications les plus connues sont par exemple Drync, Vivino ou Delectable wine, dont les logos se trouvent sur la page suivante.

Image 2 : Vivino

Image 3 : Drync

Image 4 : Delectable Wine

Les applications sont toutes construites sur le même schéma, en effet, elles se ressemblent toutes dans leur architecture. Nous prendrons l'exemple de Vivino pour la présentation.

L'écran d'accueil se compose à la manière des réseaux sociaux. En effet, l'utilisateur a possibilité de « suivre » des personnes, d'interagir avec des tiers et d'avoir des informations concernant les vins mis en vedette. Le deuxième écran permet d'accéder au « profil » de l'utilisateur afin de voir les bouteilles qu'il a scannées.

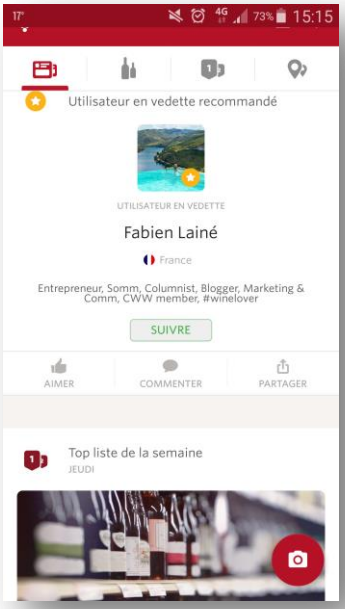


Image 5 : Vivino - Ecran d'accueil

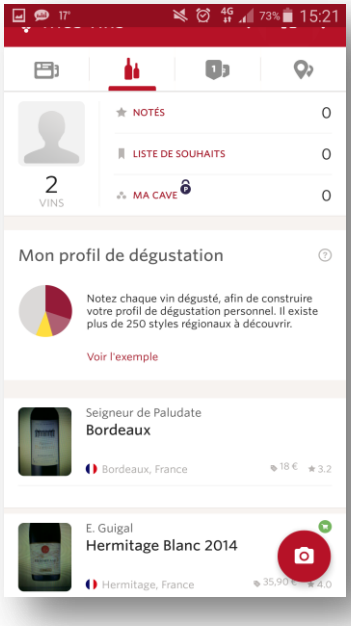


Image 6 : Vivino - Liste des vins consultés

La troisième page permet d'accéder à une boutique qui nous offre la possibilité d'acheter du vin avec les conseils et recommandations de l'application.

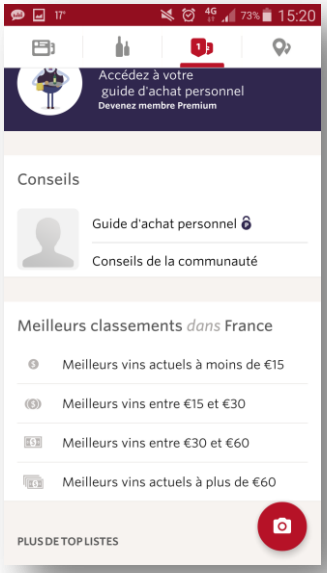


Image 7 : Vivino - Ecran d'achats

Sur toutes les pages présentées ci-dessus, nous pouvons voir un bouton « appareil photo » dans la partie inférieure droite. Ce bouton permet d'accéder à la capture de photo. L'écran de la page de capture est constitué du flux vidéo en fond, sur lequel se trouve un cadre. C'est dans ce carré que l'utilisateur doit cadrer l'étiquette de sa bouteille de vin. Une fois que la capture est réalisée, une fenêtre permet de vérifier la qualité de l'image prise et de valider l'envoi. Nous pouvons ensuite obtenir les différentes informations.

Image 8 : Vivino - Ecran de capture

Image 9 : Vivino - Résultats de la capture

Problématique

Ces applications sont fonctionnelles mais elles présentent un problème. Pour faire une analyse d'étiquette, il est nécessaire de prendre une photo, de demander à l'utilisateur de vérifier que l'image est correcte et enfin d'envoyer l'image au serveur pour analyse. Cette démarche prend du temps et demande d'avantage d'interaction de la part de l'utilisateur.

Nous souhaitons mettre en place un système novateur qui n'impose pas à l'utilisateur de prendre une photo. En effet, nous souhaitons que l'utilisateur pointe l'appareil photo du système sur une bouteille de vin. Le flux vidéo s'affichera alors et la détection de l'étiquette se fera en direct et sera affichée sur le flux vidéo de l'appareil photo. Le système prendra alors une photo dans le cas où la bouteille est correctement positionnée.

Contexte du scanner à vin

Le projet d'application de détection de bouteille est créé dans le cadre d'un projet plus large mené par le laboratoire d'informatique de Polytech Tours. En effet, le laboratoire d'informatique souhaite mettre en place un scanner à vin innovant ne nécessitant pas de prendre une photo.

Image 10 : Laboratoire d'informatique

Le projet qui m'est confié représente une partie de ce projet global. En effet, je dois créer une application qui prenne en charge la partie détection de l'étiquette. Je ne m'occuperai pas de la partie gestion d'information des bouteilles.

Périmètre du projet

Le projet que je vais mettre en œuvre fera partie du projet de scanner à vin. Il se limitera cependant à la partie détection d'étiquettes.

Mon projet se concentrera sur la partie Intreface Homme Machine (IHM) et sur la détection de l'étiquette de la bouteille de vin. Elle ne contiendra pas la partie comparaison de l'étiquette avec la base de données et renvoi des informations de la bouteille de vin.

Mon projet et le cahier des charges

Dans cette partie je vais vous présenter plus précisément le projet de détection d'étiquette. Je vais vous faire découvrir les fonctions principales de l'application et l'architecture générale de fonctionnement. Ensuite, je vous montrerai l'application que je souhaite réaliser ainsi que les contraintes de développement.

Fonctions principales

Le rôle principal de l'application que je vais mettre au point est de pouvoir permettre à l'utilisateur de détecter une étiquette sur une bouteille de vin. Pour cela nous allons devoir mettre en œuvre plusieurs fonctions principales.

- **Récupération du flux vidéo** : Cette fonction a pour rôle d'ouvrir la connexion avec la caméra et de récupérer le flux vidéo qui sort du capteur optique
- **Affichage du flux vidéo brut :** Cette fonction a pour but d'afficher le flux vidéo sortant du capteur sans aucun changement, on obtiendra alors l'affichage du flux vidéo en « temps réel » qui affiche instantanément ce que filme l'utilisateur
- **Traitement du flux vidéo :** La fonction « traitement du flux vidéo » a pour but de modifier le flux vidéo qui sort du capteur afin de rendre la fonction « Détection de l'étiquette » plus efficace
- **Détecter l'étiquette :** La fonction « Détecter l'étiquette » aura pour fonction de détecter les contours de l'étiquette sur les images du flux vidéo modifié fournis par la fonction précédente
- **Ajout de la détection sur l'image :** Le rôle de cette fonction est de rajouter la détection de l'étiquette à l'affichage qui est sur l'application. Cet affichage prendra la forme d'un carré de couleur qui s'affichera au niveau des contours de l'étiquette
- **Détection d'une « bonne » image :** Cette fonction a pour but de trouver la photo la plus optimale, elle donne ainsi le signal à la fonction « Extraction et stockage de l'image » de « prendre une photo »
- **Extraction et stockage de l'image :** Cette fonction a pour rôle d'extraire une image du flux vidéo lorsque la fonction précédente lui indique, l'image sera alors stockée dans la mémoire du terminal.
- **Interface homme machine :** Il sera nécessaire de mettre en place une interface homme machine qui permettra à l'utilisateur d'utiliser l'application
- **Visualisation des photos prises :** Une fonction qui m'a semblé intéressante à mettre en œuvre est la visualisation des photos extraites grâces à l'application.

Architecture de la partie détection d'étiquettes

La partie de détection d'étiquette peut se représenter la manière suivante :

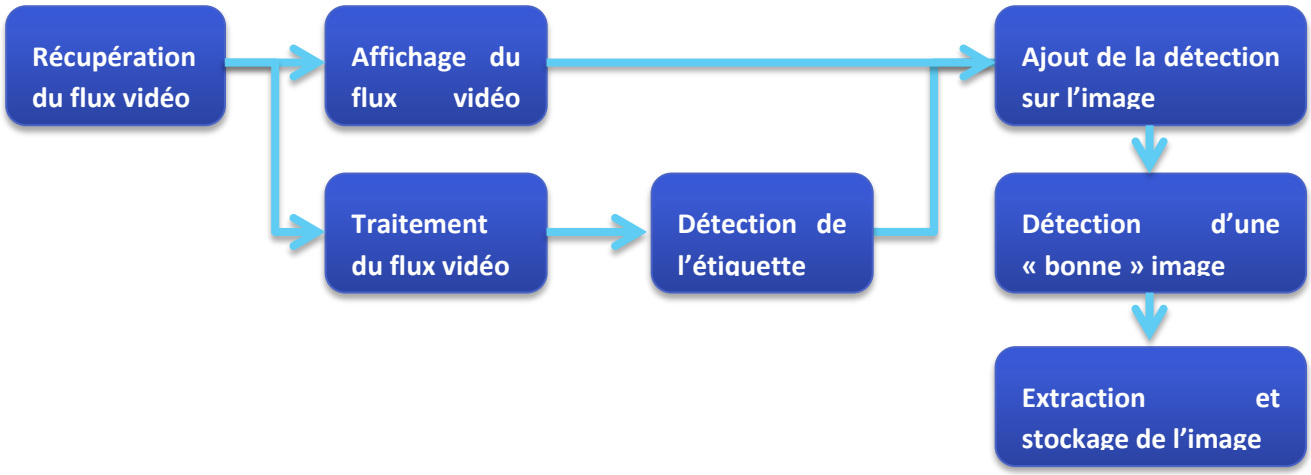


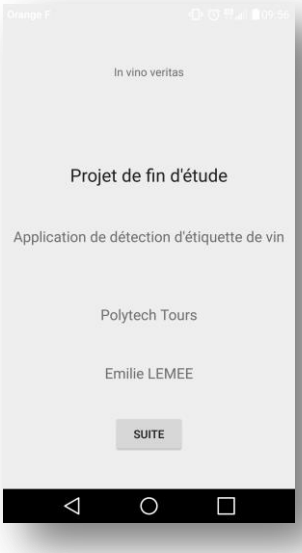
Image 11 : Schéma de la détection de l'étiquette

La première étape consiste à obtenir le flux vidéo de l'appareil photo. Il faut ensuite l'afficher le flux vidéo. En parallèle, nous traitons le flux vidéo et détectons l'étiquette. Une fois l'étiquette détectée, on l'ajoute sur l'image. Ensuite on détecte une « bonne image » et on extrait et on stocker l'image.

Présentation de l'application

L'application devra permettre à l'utilisateur d'utiliser le système de détection d'étiquette. Pour cela l'application comportera plusieurs « écrans ». J'ai créé quatre écrans :

› Ecran d'accueil



C'est l'écran qui s'affiche lorsque l'application s'ouvre. Il permet de donner les informations à l'utilisateur sur l'application.

Image 12 : Ecran d'accueil

› Ecran principal

Nous arrivons ensuite sur la page principale de l'application. Cette page a pour but de rediriger l'utilisateur vers les deux autres fenêtres qu'il peut atteindre. Elle est donc constituée de deux boutons.

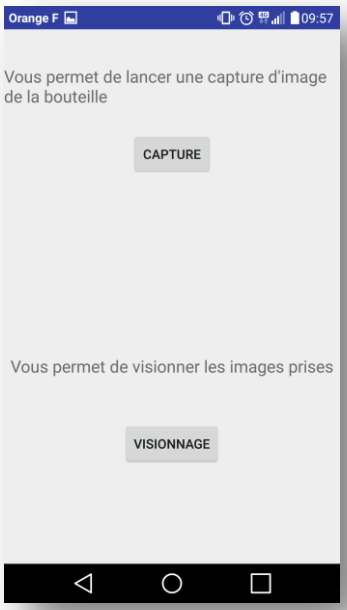
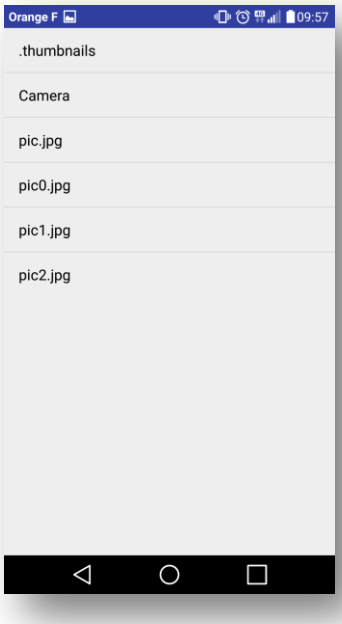


Image 13 : Page principale

› Ecran de visionnage



La page de visionnage permet de voir les différentes photos qui ont été prises lors du l'utilisation de l'application.
Un clic sur l'image nous permet d'afficher l'image prise par l'application.

Image 14 : Page de visionnage

› Ecran de capture

La dernière page est accessible depuis l'interface principale en cliquant sur le bouton « Lancer une capture ». Cette interface possèdera un écran sur lequel sera affiché notre flux vidéo. Il y aura une forme (carré ici) qui mettra en avant la détection en l'entourant. Il y aura aussi un bouton qui permettra de lancer une nouvelle capture.

›

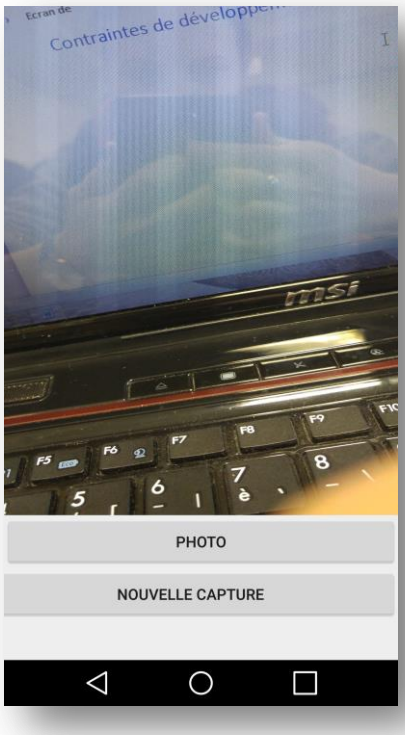


Image 15 : Page de capture

L'enchainement des pages de l'application se fait de la manière suivante :

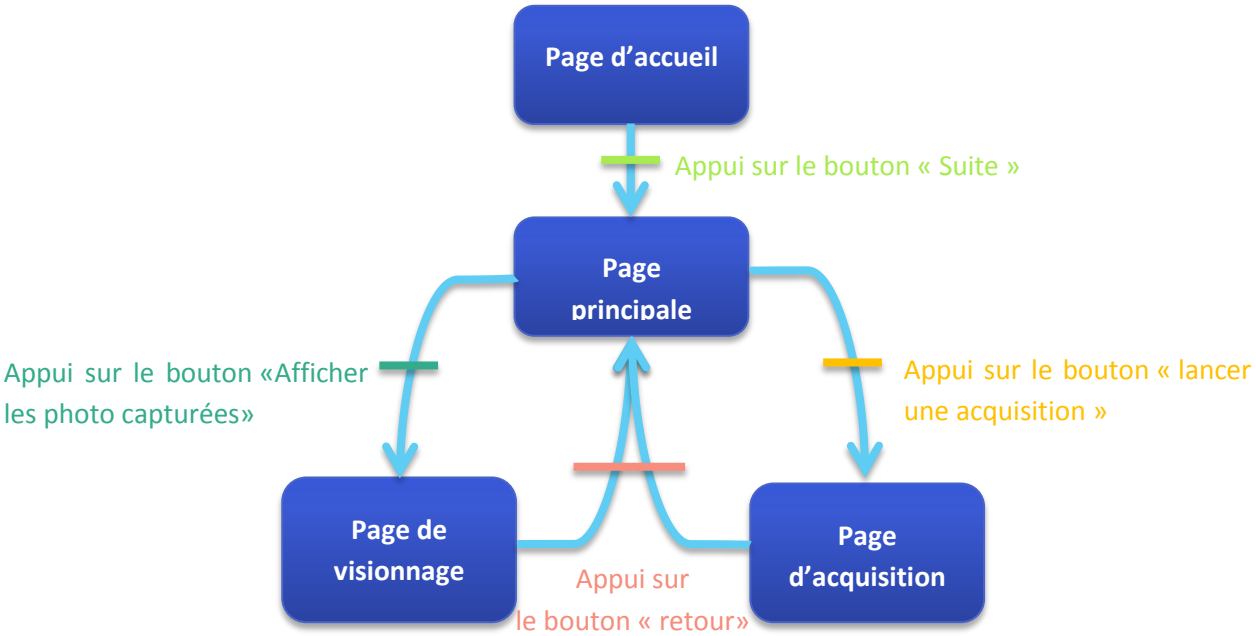


Image 16 : Enchainement des écrans

Contraintes de développement

Nous allons étudier les différentes contraintes de développement qui ont été données pour la mise en place de ce projet.

Les cibles

L'application de détection de bouteille de vin a pour but d'être utilisée sur des terminaux mobiles. Il était imposé dans le sujet de cibler les terminaux mobiles ayant un système d'exploitation Android. Pour développer l'application, je choisirai donc l'environnement de développement officiel : AndroidStudio.

Dans le sujet du projet, l'utilisation du package « hardware.camera2 » est imposée. Ce package est disponible sur les cibles Android récentes car il nécessite la version Android 5.01 (Lollipop) au minimum. Cette version correspond à l'API 21 d'Android. C'est l'API minimum avec lequel nous travaillerons.

L'application

Afin d'avoir un rendu fluide de l'application, l'affichage doit être quasi instantané. Il faut donc que les calculs soient rapides à mettre en œuvre afin de ne pas créer de retard entre l'application et la réalité.

Cette application aura pour but d'être utilisée sur des terminaux mobiles de types Android. Nous avons alors des contraintes de systèmes embarqués. Je devrai, par exemple, créer un programme qui ne soit pas trop « gourmand » en ressource et en énergie.

La méthode pour la détection de l'image est fourni par l'encadrant : Monsieur DELALANDRE.

Cahier de spécifications

Pour plus d'informations sur ce sujet, je vous propose d'aller consulter le cahier de spécifications rendu le 15.11/2015 sous le nom « Détection détiquette de bouteille de vin - Cahier de spécifications - LEMEE Emilie.pdf ».

Etude et analyse

Dans cette partie, nous aborderons les études auxquelles j'ai réflechi pour réaliser le projet. Je vais d'abord vous présenter le fonctionnement global de la détection d'étiquettes. Ensuite nous verrons le premier étage de la détection que j'ai mis en œuvre et enfin les choix techniques que j'ai décidé de mettre en place.

Fonctionnement de la détection

La détection qui est mise en place pour la détection d'étiquettes repose sur l'étude du contraste d'une image. En effet, pour la plupart des bouteilles de vin rouge, la bouteille est sombre et l'étiquette est de couleur claire.

Afin d'avoir une détection efficace et rapide, il a été décidé de découper la détection en trois étages différents. En effet, pour la même efficacité de détection, la complexité d'un algorithme seul sera beaucoup plus élevée que la complexité de trois étages différents. Le premier étage va permettre de détecter les points de l'image qui présentent un fort contraste, le reste des étages va permettre d'affiner cette sélection de points pour les centrer autours de l'étiquette. Nous aurons donc le schéma suivant :

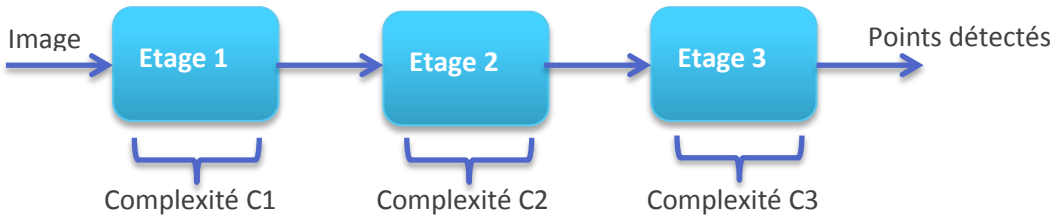


Image 17 : Schéma pour la complexité

On a C1<<C2<<C3. Il est alors plus simple de départager la détection en plusieurs étages.

Pour mon projet, il est demandé de mettre en œuvre le premier étage de la détection d'étiquette. Le deuxième étage a déjà été codé par Monsieur DELALANDRE.

Premier étage de détection

Pour mon projet, je ne mettrais en place que le premier étage de la détection d'étiquettes. Cela permettra d'avoir un premier nuage de points qu'il faudra ensuite afficher.

La méthode que nous allons appliquer est la suivante :

- Nous allons choisir un pas Δ=10pixels par exemple
- On parcourt l'image tous les Δ pixels
- On fait le calcul du point moins celui d'avant $|I1 - I2|$ on obtient la valeur α
- On fixe un seuil β
- Si $\alpha<\beta$ on considère que le point est à retenir, il est sur le contour de l'étiquette

› Si α>β on considère que le point n'est pas à retenir
› On stocke le résultat dans un tableau
› On affiche le résultat sur l'image

On peut le représenter de la manière suivante :

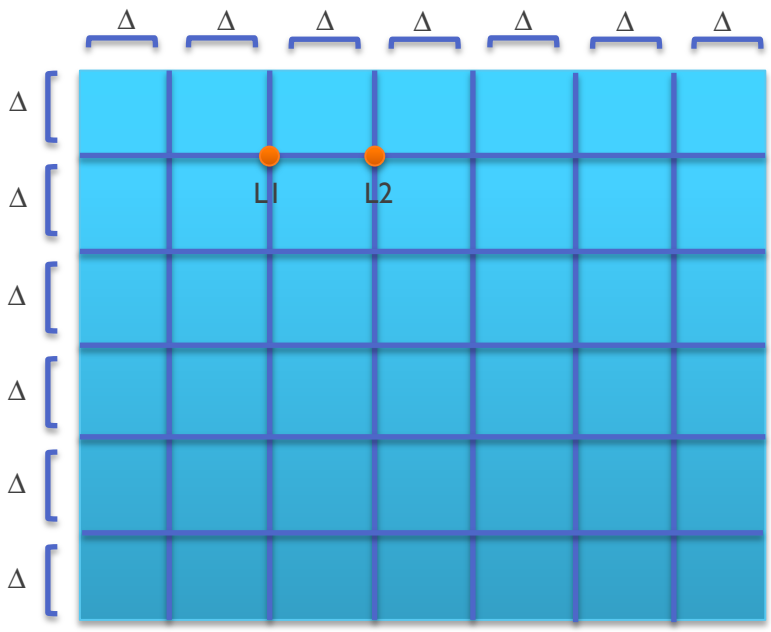


Image 18 : Explication de la détection

On créera un tableau dans lequel on stockera le résultat de l'opération. Il nous suffira ensuite d'afficher les points contenus dans le tableau sur l'image.

Enregistrement des images

L'enregistrement des images se fera par session. L'utilisateur lance la capture, une nouvelle session est créée. Lorsque l'utilisateur va prendre une photo, l'application va créer un fichier « pic0 ». Si l'utilisateur prend de nouvelles photos, l'image sera écrasée.

L'utilisateur va ensuite créer appuyer sur le bouton « Nouvelle capture », l'application va donc lancer une nouvelle session, lorsque l'utilisateur prendra une nouvelle image, un nouveau fichier « pic1 » est créé et c'est ce fichier qui sera écrasé lors de futures prises de photos.

Cahier d'analyse

Pour plus d'information sur l'analyse et la modélisation, je vous propose d'aller consulter le cahier d'analyse et modélisation rendu le 03/01/2016 sous le nom « Détection d'étiquette de bouteille de vin - Cahier d'analyse et modélisation - Emilie LEMEE.pdf ».

Réalisation et gestion de projet

Dans cette partie je vais vous présenter le déroulement de la réalisation de l'application. Nous allons voir de l'apprentissage du langage Android, ainsi que les différentes étapes dans la mise en œuvre de mon projet.

Formation

Le langage Android est un langage nouveau pour moi. En effet, je n'avais utilisé Android que lors d'un projet (avant lequel je n'avais jamais codé ni en Java, ni en Android) ainsi que lors des cours donnés cette année. J'ai choisi ce projet, dans le but de progresser dans ce domaine. J'ai donc choisi de consacrer une période de mon projet à l'apprentissage et la formation.

Pendant cette période de formation, j'ai abordé le sujet des applications à plusieurs fenêtres (utilisation d'intents etc..) mais, je me suis surtout concentrée sur l'utilisation de l'appareil photo (package hardware.camera2) que je ne connaissais pas du tout.

Affichage du flux vidéo et extraction des images

La première étape sur laquelle j'ai choisi de travailler est la partie affichage de vidéo ainsi qu'à l'enregistrement des images sur la mémoire du terminal mobile.

Flux vidéo

La première étape du projet a été la récupération du flux vidéo de l'appareil photo.

Pour cela, il faut commencer par configurer la caméra et ouvrir la connexion. Nous utilisons la fonction openCamera qui est la suivante :

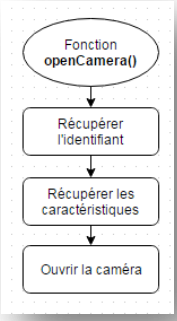


Image 19 : Fonction openCamera

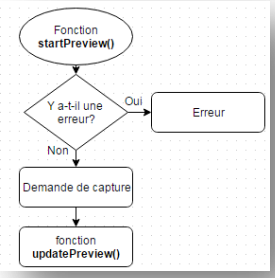


Image 20 : Fonction startPreview

Une fois que la caméra est ouverte, à l'issu de la fonction « openCamera », un évènement « CameraOpened » se déclenche. C'est dans cet évènement que l'on va lancer la visualisation avec la fonction « startPreview » ci-dessus.

« startPreview » appelle la fonction « updatePreview » dont le rôle est de mettre à jour l'affichage de la caméra, elle se présente comme suit:

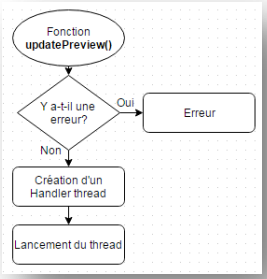


Image 21 : Fonction updatePreview

Extraction des images

Sur la partie graphique, nous ajoutons un bouton « Photo » sous l'objet « TextureView ». Dans le code, je créée un « listener » sur ce bouton, afin de surveiller l'appui sur ce bouton.

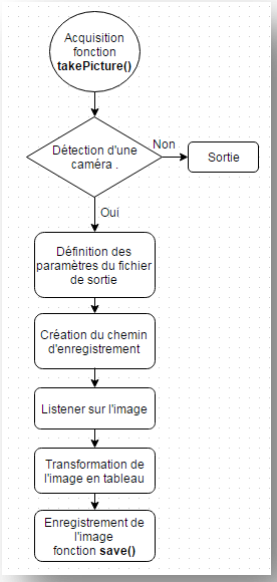


Image 22 : Algorithme de la fonction d'acquision

Cette fonction est appelée lors de l'appui sur le bouton « photo ». Elle permet, lorsqu'il y a une caméra, de définir les paramètres du fichier que l'on souhaite enregistrer (taille, type…). On créée ensuite notre chemin. On va enfin transformer l'image pour pouvoir l'enregistrer puis, on appelle la fonction save.

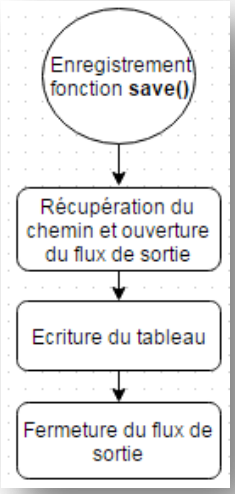


Image 23 : Algorithme de la fonction save

Cette fonction permet d'enregistrer l'image transformée par la fonction takePicture(). Elle écrit le tableau sur la mémoire de la carte SD.

Nous choisissons le chemin suivant : `Environment.getExternalStorageDirectory()+"/DCIM", "pic.jpg"`. C'est-à-dire qu'on va enregistrer l'image sur la carte mémoire externe, dans le dossier « DCIM ». L'image aura le nom « pic.jpg ».

Mise en place des sessions de capture

Le système d'acquisition permet d'enregistrer une image, le problème étant que cette image est écrasée à chaque nouvelle acquisition. Nous souhaitons que l'image soit effectivement écrasée, mais l'utilisateur doit pouvoir lancer une nouvelle session et enregistrer une nouvelle image.

Je créée alors un nouveau bouton sur l'interface « Nouvelle capture ». Dans le code, je crée une nouvelle variable globale « i » que j'initialise à la valeur 0. Je créée un « listener » sur le bouton qui aura alors le rôle suivant :

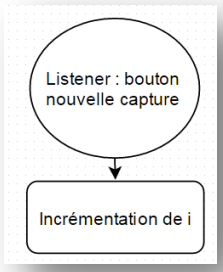


Image 24 : Algorithme du listener bouton nouvelle capture

On change alors le chemin d'enregistrement par le suivant :

```
Environment.getExternalStorageDirectory()+"/DCIM", "pic"+i+".jpg"
```

La première image enregistrée sera alors « pic0.jpg ». Si l'utilisateur appuie sur le bouton « Nouvelle capture », une image « pic1.jpg » sera créée.

Manipulation du buffer

Afin d'utiliser l'image que nous obtenons, nous devons manipuler le buffer afin d'appliquer la détection d'étiquette de manière plus rapide. Le flux vidéo de base provenant de la caméra Android est YUV. C'est-à-dire que l'image est découpée en trois composantes Y : la luminance (image en nuance de gris), U et V : des composantes en couleur. Voici un exemple :

Image 25 : Image originale

Image 26 : Composante Y

Image 27 : Composante U

Image 28 : Composante V

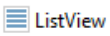
Il nous faut donc récupérer la composante Y sous forme de tableau d'octets.

Nous souhaitons faire des calculs sur cette image en nuances de gris fournis par la composante Y de notre image. Nous devons donc extraire cette composante de l'image de manière régulière afin d'effectuer les calculs dessus.

Pour l'extraction, nous utiliserons le même principe de base que pour l'extraction des images permettant de prendre des photos. En effet, nous transformons l'image en tableau d'octets et c'est sur ce tableau de nous allons travailler pour détecter l'étiquette.

Visionnage des fichiers

J'ai ensuite souhaité mettre en place un système permettant de visualiser les images qui ont été prises par la partie d'acquisition. J'ai donc créé une nouvelle page qui me permettra de choisir le chemin répertoire dans lequel nous verrons les images. Il me faudra aussi pouvoir les ouvrir. Je crée donc une nouvelle classe pour la page de visionnage : PageVision.

Pour créer la partie graphique de l'application, je vais uniquement utiliser un objet ListView : 

. Cet objet permet de créer une liste verticale déroulante.

Pour la partie code, je vais vous présenter le fonctionnement de l'application sous forme de schéma :

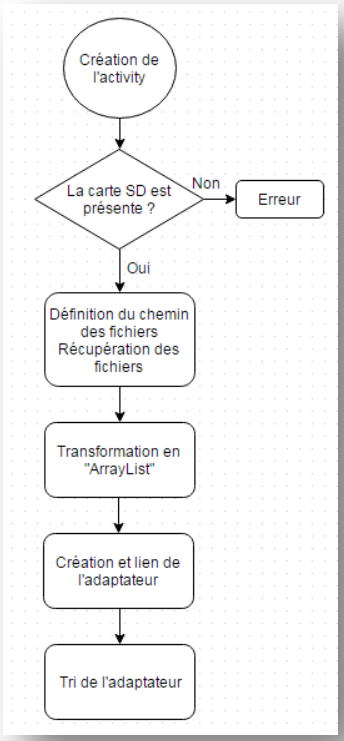


Image 29 : Algorithme lors de la création de l'activity

Lors de la création, l'application va vérifier la présence de la carte SD (sur laquelle sont enregistrées les images). Ensuite, elle va effectuer une série d'action qui va permettre d'afficher les informations du répertoire de base où elle se trouve (répertoire où se trouvent les images capturées).

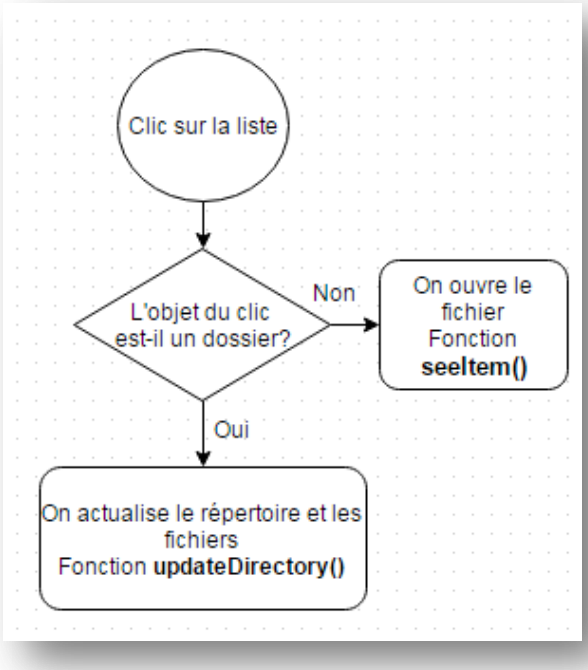


Image 30 : Algorithme lors du clic sur la liste

Lors d'un clic sur la liste, on va vérifier le type de l'objet (image ou répertoire). Le programme va alors actualiser l'affichage pour un répertoire ou lancer le visionnage de l'image pour le fichier.

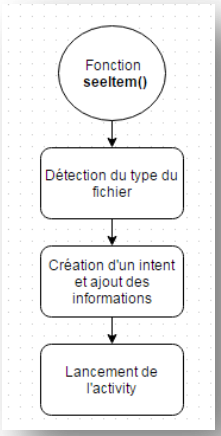


Image 31 : algorithme de la fonction seeItem

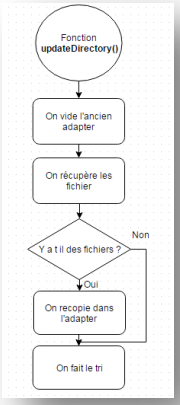


Image 32 : Algorithme de la fonction updateDirectory

La fonction seeItem va permettre de détecter le type de fichier. Elle créera ensuite un intent (ouvrir une nouvelle page) qui utilisera un système adapté pour l'ouvrir.

La fonction updateDirectory permet d'actualiser le répertoire. Quand l'utilisateur clique sur un répertoire dans la liste de fichier, il faut modifier le chemin pour entrer dans le fichier, effacer les anciens fichiers affichés et afficher les nouveaux.

Interface Homme Machine

Après avoir fait fonctionner les deux entités suivantes indépendamment, j'ai choisi de mettre en place l'IHM de mon application.

Page d'accueil

La page d'accueil est construite de manière très simple. Elle affiche uniquement les informations concernant le projet et contient un bouton permettant de passer à la page suivante.

› Affichage graphique

On n'utilisera uniquement que des objets graphiques de type Ab Plain TextView dans lequel nous entrons le texte de présentation. Nous utiliserons un OK Button pour passer à la suite.

› Code

Le code est aussi très simple. Il ne contient que l'affichage de la fenêtre graphique : `setContentView(R.layout.aff_page_accueil);`, ainsi que la mise en place d'un « listener » sur le bouton « Suite ». Cette action nous permet d'exécuter une action lors de l'appui du bouton, ici nous lancerons la page suivante : la page principale. Nous utiliserons un « intent ».

Page principale

La page principale nous permet de naviguer vers la page de visionnage ou vers la page de capture d'image.

› **Affichage graphique**

De même que pour la page d'accueil, nous utiliserons uniquement des objets « TextView » dans le but d'afficher les informations sur les boutons. Nous utiliserons deux boutons pour lancer la page de visionnage et de capture.

› **Code**

Le code est composé de la partie affichage : on affiche la partie graphique (même procédure que la page d'accueil), ainsi que de la partie changement de page. Nous créons un « listener » par bouton. Pour chaque bouton, nous allons créer un « intent » pour lancer la page correspondante.

Page de visionnage

Le visionnage permet à l'utilisateur d'avoir accès aux fichiers créés par l'application. C'est dans cette page que nous intégrons la partie visionnage qui a été présentée précédemment.

› **Affichage graphique**

L'affichage de la liste se fait grâce à un objet « ListView ». Il est le même que celui présenté dans la partie « **Visionnage des fichiers** ».

› **Code**

Le code présent dans cette page est celui présenté dans la partie « **Visionnage de fichiers** ».

Page de capture

La page de capture permet à l'utilisateur de faire des captures d'images de l'application. L'utilisateur peut en effet voir le flux vidéo de l'application, capturer une image, ou lancer une nouvelle capture.

› **Affichage graphique**

L'affichage du flux vidéo se fait grâce à un objet « TextureView » sur lequel on voit apparaitre les images de l'appareil photo. J'utilise aussi deux boutons : « Nouvelle capture » et « Photo ». L'interface est la même que celle présentée dans la partie « **Affichage du flux vidéo et extraction des images** ».

› **Code**

Le code présent dans cette page est celui présenté dans la partie « **Affichage du flux vidéo et extraction des images** ».

Détection de l'étiquette

J'ai choisi de mettre en place la détection d'étiquettes sur une application différente le temps du débuggage. Une fois que l'application fonctionnera parfaitement, je l'intégrerai au corps de l'application existante.

Nous allons partir d'un tableau de valeur représentant une image en noir et gris (résultat de la manipulation de buffer sur l'application principale). Nous allons donc appliquer la méthode de détection présentée précédemment sur l'image.

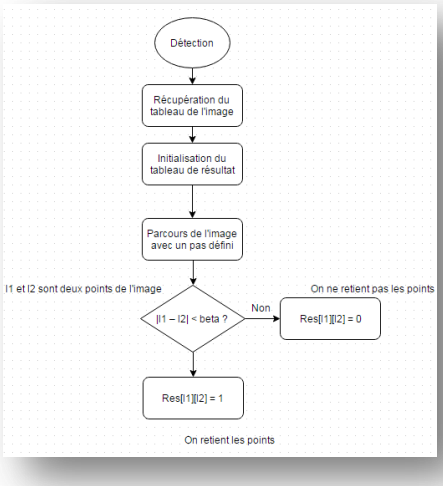


Image 33 : Algorithme de detection

On affiche ensuite les points retenus du tableau (points à 1), sur l'image en faisant la moyenne des coordonnées I1 et I2.

Livrables

Ce projet de fin d'étude était placé sous le signe de la gestion de projet. En effet, au contraire des autres projets que nous avons menés à Polytech, ce projet nécessitait plusieurs livrables.

Cahier de spécification

Le cahier de spécifications est le premier livrable que nous avons rendu. Dans ce dossier, nous présentons le projet que nous avons choisi, nous présentons divers éléments tels que le contexte du projet, les fonctions demandées et toutes les données générales du projet. Nous présentons aussi les premières pistes d'analyse que nous avons étudiées, ainsi qu'un planning prévisionnel.

Analyse et modélisation

Le cahier d'analyse et modélisation est le deuxième livrable à présenter, il nous permet de définir l'analyse du projet que nous avons faite à ce moment. Sa date de rendu est située environ au 2/3 du projet, mais il est voué à évoluer et une version final sera à rendre à la fin du projet.

Dans ce dossier nous présentons l'analyse du projet qui nous est donné de faire. Nous présentons tous les éléments que nous souhaitons mettre en place pour le développement de notre application.

Reprise et utilisation

Les livrables de reprise et utilisation sont composés de plusieurs éléments, ils sont à rendre en fin de projet. Leur rôle est de permettre à la personne qui va reprendre le projet de pouvoir le maintenir et l'utiliser. Pour mon projet, il a été décidé qu'il était nécessaire d'avoir deux livrables : un guide d'utilisateur, ainsi qu'un guide pour la maintenance.

Le guide d'utilisateur comprendra les différentes informations qui permettent à l'utilisateur de faire fonctionner l'application, guide d'installation, d'utilisation.

Le guide de maintenance est destiné au programmeur qui reprendra le projet. On y trouvera la structure du projet, les différentes analyses et modélisations ainsi que les procédures de test.

Planning prévisionnel et réel

Afin d'avoir une vision plus clair du planning prévisionnel et réel, je l'ai découpé en deux parties : les livrables, et la programmation.

Le premier planning, présenté ci-dessous correspond aux différents livrables. Les barres grisées représentent le planning prévisionnel qui a été mis au point en début de projet. Les barres de couleurs représentent le planning réel qui a eu lieu.

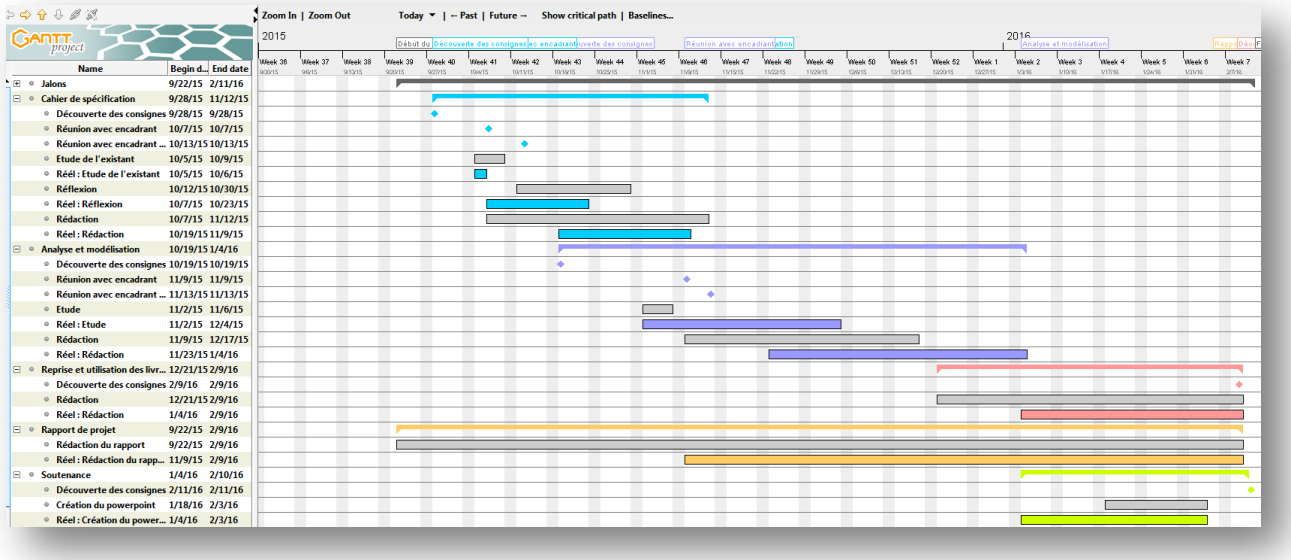


Image 34 : Planning des livrables

Le planning présenté ci-dessous correspond au déroulement de la programmation de l'application. De la même manière que le diagramme précédent, les barres grisées représentent le planning prévisionnel, les barres de couleur représentent l'avancée réelle.

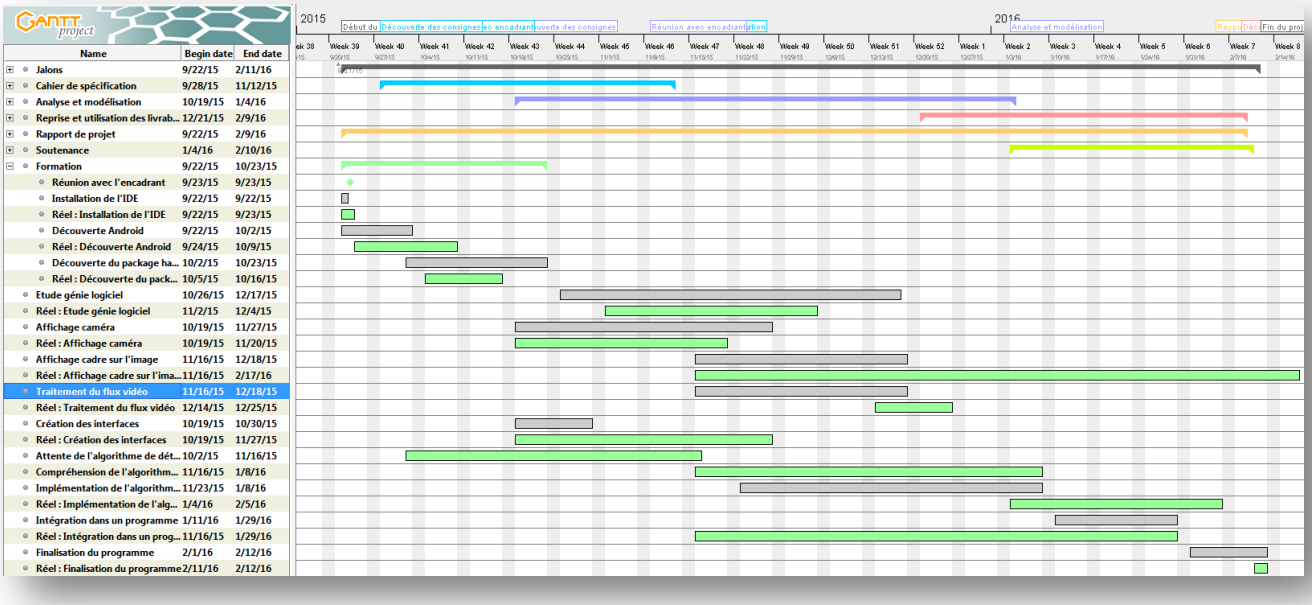


Image 35 : Planning de programmation

Point sur le projet

Le projet est sur le point d'être terminé, la partie IHM est complète, il est possible de naviguer dans l'application et d'utiliser les différentes pages de l'application. La partie visionnage est aussi terminée, nous pouvons visualiser les images capturées par l'application.

Sur cette application, il est aussi possible de visualiser le flux vidéo de l'appareil photo, ainsi que de prendre des photos avec le système de « session » et de les enregistrer sur la mémoire de l'appareil.

J'ai choisi de travailler sur la détection de l'étiquette dans une nouvelle application (à part) afin de simplifier la mise en œuvre et d'éviter de poser problème sur la partie IHM qui fonctionne. Cette application permet, à partir d'une image en noir et blanc de traiter le contraste. Cette application est encore en cours de développement, je modifierai le document une fois cette étape terminée.

Bilan

Ce projet Android m'aura permis de m'améliorer en programmation, notamment en langage Android. J'ai donc dû prévoir une étape de formation. Le développement de l'application m'a permis d'utiliser des outils nouveaux pour moi, tels que le package contrôlant la caméra du smartphone etc… Ce projet m'aura donc permis d'approfondir ce langage de programmation.

Au cours du projet, j'ai rencontré quelques difficultés. La majeure partie de ces difficultés est survenue lors de la programmation. En effet, n'ayant que peu de connaissances en Android, j'ai été confronté à des blocages (bugs ou autres problèmes). J'ai donc dû utiliser internet dans le but de trouver des exemples qui m'aideraient à résoudre mes problèmes. Cela m'a aussi permis d'acquérir des automatismes de programmation dans ce langage.

La gestion du temps était un problème assez important dans ce projet. En effet, bien que du temps nous ait été accordé pour travailler, nous avions plusieurs livrables à rendre nous demandant un travail assez conséquent. Il a donc été assez difficile de répartir le temps entre la réalisation du projet et la rédaction des documentations.

La fin du projet approchant, je peux prendre un peu de recul vis-à-vis de la réalisation de mon projet. Je pense ne pas avoir communiqué suffisamment avec l'encadrant du projet. En effet, lorsque je rencontrais un problème, il aurait pu m'orienter sur une solution me permettant d'avancer avec moins de « blocages ».

Conclusion

Le projet de fin d'étude que j'ai choisi de réaliser concerne la réalisation d'une application Android pour la détection d'étiquettes de bouteille de vin. Ce projet s'intègre dans un projet de scanner à vin créé par le laboratoire d'informatique de Polytech Tours.

Mon choix s'est porté sur ce projet car, n'ayant que très peu de connaissances en Android, je souhaitais approfondir ce domaine de programmation. Les smartphones et tablettes font partie intégrante de nos vies personnelles, mais s'immiscent aussi dans la vie professionnelle comme nous pouvons le voir au quotidien.

Ce projet m'a permis d'acquérir de nombreuses connaissances dans ce domaine, ainsi que des automatismes de programmation. En effet, j'ai pu découvrir l'utilisation de divers objets (TextureView, ListView), améliorer des systèmes vus en cours (système d' « intents »), ainsi que de manipuler de nouveaux packages (hardware.camera2 …).

Je regrette de ne pas avoir eu le temps de terminer mon projet et ferai de mon mieux pour présenter un logiciel fonctionnel lors de la soutenance qui aura lieu le 09/02/15 à 09h05.

Table des images