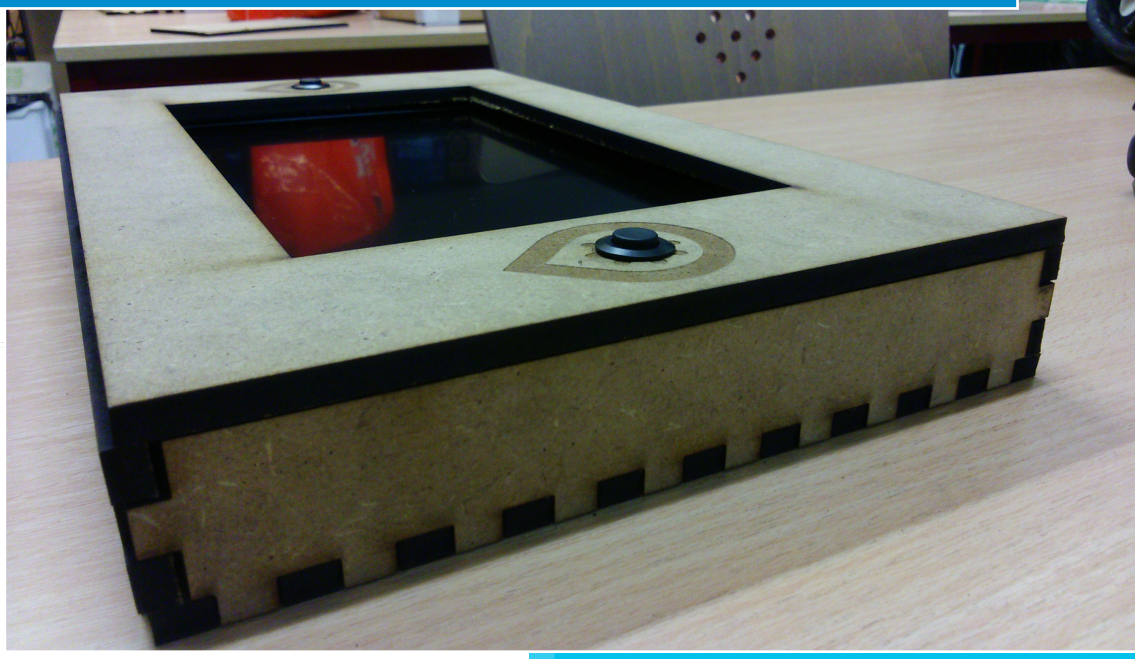
POLYTECH LILLE

Université de Lille 1 SCIENCES ET TECHNOLOGIES

Rapport de Projet S8

Partage simplifié – Boite documentaire



Année Universitaire 2014/2015
Projets IMA4 SC

Elèves
Julien HERIN
Jérémie DENECHAUD

Encadrants école
Alexandre BOE
Thomas VANTROYS
Xavier REDON
Rodolphe ASTORI

Table des matières

Introduction

Ce rapport a pour but d'exposer le travail réalisé dans le cadre du projet IMA4 SC du semestre 8.

Nous avons choisi le sujet obligatoire "Partage simplifié". Ce choix a été motivé par la volonté de participer activement au développement du nouveau Fabricarium, inauguré en octobre 2014. L'élaboration d'un outil qui serait utilisé régulièrement dans cet espace était un défi que nous voulions relever. Nous avons aussi beaucoup d'affinités avec la programmation de systèmes communiquant et les nouvelles technologies, et nous voyons dans ce sujet l'occasion d'exprimer assez librement notre créativité.

Nous commencerons tout d'abord par présenter le contexte du projet et par décrire le cahier des charges que nous nous sommes fixé. Nous parlerons ensuite de la partie software puis hardware en évoquant à chaque fois le résultat obtenu, les problèmes rencontrés, et les éventuelles possibilités d'amélioration.

1 Cahier des charges

1.1 Contexte

Polytech Lille accueille depuis octobre 2014 un nouvel espace de création, "le Fabricarium". Ce FabLab est un atelier ouvert où chacun peut venir et utiliser des outils plus ou moins avancés pour créer des projets professionnels ou personnels. Pour respecter la charte des FabLab, il est nécessaire que chaque projet soit un minimum documenté.

1.2 Objectifs du projet

Notre projet, la "boite documentaire" du Fabricarium doit dans un premier temps permettre à tous ceux qui le veulent de documenter rapidement et simplement son projet.

Dans un second temps, pour les utilisateurs qui le veulent, une plate-forme plus avancée peut être mise à disposition pour modifier la documentation de façon plus personnalisée.

Cette boite documentaire doit être la plus simple d'utilisation pour pouvoir partager simplement ses réalisations. La cible est fixée de 7 à 77 ans, avec des contraintes fortes incluant les technophobes, à qui la vue d'un clavier fait tourner la tête.

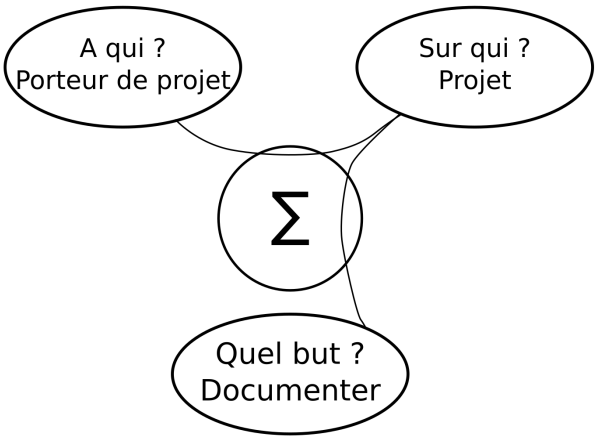


Figure 1 : Diagramme "bête à cornes décrivant le système"

1.3 Définition du besoin

Cette boite documentaire doit permettre de simplement :

- Prendre une vidéo d'une personne désirant expliquer sa réalisation
- Prendre des photos des objets liés à son projet
- Ajouter une légende
- Publier le "reportage" réalisé sur différentes plates-formes à l'aide d'un simple bouton

Après une réunion avec des utilisateurs du Fablab, le besoin a été légèrement redéfini. Pour la majorité des utilisateurs, (i.e. d'autres étudiants de Polytech Lille), la solution la plus simple est d'avoir une application pour Smartphones. Ainsi toutes les photos et vidéos peuvent être facilement capturées et uploadées. Cependant, comme tout le monde ne possède pas un Smartphone dernier cri avec une qualité d'appareil photo suffisante, un outil accessible a tous est tout de même nécessaire.

2 Partie Software

2.1 Travail réalisé

La partie “software” de notre projet comprend l’ensemble des solutions adoptées pour traiter les informations récupérées par les boutons de la boite. Ceci inclût :

- Un programme principal écrit en C
- La gestion d’interface graphique (librairie SDL)
- La gestion du flux vidéo de la webcam (librairies Video4Linux2, libjpeg-turbo-devel et libtheora-devel minimum)

D’autre part, un site web basique a été réalisé pour stocker les photos et vidéos.

2.1.1 Le programme principal

Le programme **“main”** gère entre autres la scrutation des boutons. Sur la BeagleBone, on accède aux dossiers **gpio66** à **gpio68** et **gpio44** correspondant aux quatre boutons pour y lire les fichiers “**value**”. Ces fichiers contiennent la valeur 1 si le bouton est enfoncé, et 0 s’il est relâché.

utils.c contient des fonctions qui permettent de créer différents cas selon l’état du bouton. Par exemple la fonction ***updateVal*** permet de mettre a jour la valeur lue de chaque bouton et de stocker l’ensemble des informations dans un *int*. Les actions des boutons sont différentes selon l’état précédent de la boite. Tout ceci est géré dans un grand *switch/case* (presque 50% du **main.c**). Le programme boucle infiniment sur ce *switch/case* et attend une interaction de l’utilisateur. L’utilisation d’un flag ainsi que la fonction ***usleep*** permettent au programme d’éviter de repasser involontairement dans le même *case* quand l’utilisateur reste appuyé sur le même bouton. Tous les boutons doivent être relâchés pour remettre le flag à 0.

Un code de retour a été programmé : lors de l’appui des deux boutons simultanément, on sort de la boucle et on quitte le programme *(Figure 2)*.

```
case BOTH:
        if ( !flag )
        {
                flag = 1;
                SDL_FreeSurface(backgroundPicture);
                SDL_Quit(); /* Stop SDL & free memory */
                return EXIT_SUCCESS;
        }
        break;
```

Figure 2 : Code de retour du programme principal

2.1.2 Fonction auxiliaire

Lors du passage dans le cas adapté, ce programme appelle la fonction ***webcam*** qui est définie dans *./cam/hwebcam.h*.

*void webcam(bool capture_picture, int english, char *bouton1, char *bouton2);*

Cette fonction s'occupe uniquement d'afficher et capturer le flux vidéo de la webcam. Le booléen ***capture_picture*** vaut alors respectivement ***false*** ou ***true***. Le fichier ***hwebcam.c*** contient un code récupéré d'internet, et adapté à notre utilisation (Giorgio Vazzana / GNU GPLv3). Le ***README*** contenu dans le dossier *webcam* est celui d'origine et on peut y voir des exemples d'utilisation du programme avant modification.

A l'origine, le programme était suffisant à lui-même et permettait avec les bonnes options d'afficher et capturer le flux de la webcam dans une fenêtre. La fonction ***webcam*** était donc auparavant la fonction ***main***. Nous avons donc dû adapter tout le programme pour l'inclure dans notre environnement. Ceci implique d'écrire en dur les options qui nous sont propres et de passer en paramètres les paramètres de capture/affichage du flux. On a donc choisi le bon format des pixels, le choix d'encodage vidéo/audio. L'audio est également enregistrée. Nous avons cependant laissé le code correspondant à d'autres configurations pour faciliter une future évolution du code, un changement de webcam, etc.

Malgré un temps de latence un peu élevé, la capture d'une photo fonctionne correctement. On a également le retour visuel de notre capture, et la possibilité de recommencer si l'on veut. L'encodage se fait grâce à Theora pour la vidéo et Vorbis pour le son, mais ceci ne constitue aucunement le centre de notre projet. Comme SDL ne peut afficher que des images bitmap, il est nécessaire d'effectuer une conversion de ppm vers bmp. Ceci explique la latence obtenue.

Une fois la capture validée, le résultat est renommé proprement (date et heure), puis copié via le réseau dans le dossier */home-www/ima4/jherin12/images*. Une fois le fichier correctement transféré, on le supprime de la BeagleBone. On peut voir ces opérations dans ***main.c*** à la ligne 201 *(Figure 3)*.

```
// SCP_CMD      = "scp -P 2222 ./pics/img* jherin12@portier.polytech-
lille.fr:/home- www/ima4/jherin12/upload/images/"
```

```
// On remplit la chaine avec le format choisi
strftime(format, 128, "%Y_%m_%d__%X", &date);
sprintf(commande, "mv pics/image.bmp pics/img_%s_UTC.bmp", format);
system(commande);
system(SCP_CMD); //Defined at line 17
system("rm pics/img*");
```

Figure 3 : Opérations sur la photo enregistrée

Pour s'authentifier automatiquement au serveur de l'école, nous avons utilisé l'authentification par clé.

Pour ce qui est du choix de la langue, le choix des menus à afficher est pondéré par un *int english* qui décalera toutes les valeurs afin de choisir les menus adaptés.

Intéressons-nous maintenant au site et ce qu'il s'y passe une fois les données placées dans le bon répertoire.

2.1.3 Site web

Nous avions initialement créé le site *http://boitefablab.plil.net/ (Figure 4)* afin d'y héberger un prototype de la partie software. Celui-ci émulait la partie software sans se soucier des technologies utilisées afin de faire valider au plus vite le modèle par nos professeurs encadrant et par les utilisateurs du Fabricarium. Nous avons pu ainsi sereinement nous lancer dans la production software.

Figure 4 : Prototype en ligne de la boite

Nous avons ensuite décidé d'utiliser le même site pour tester l'upload des fichiers créés par la BeagleBone. A l'heure actuelle, on y trouve toujours un prototype qui nous permet d'utiliser virtuellement la boite et de voir la plupart des fonctions disponibles. Ce prototype a été réalisé en créant des images cliquables grâce à des images map *(Figure 5)*. Ceci permet de définir des zones qui déclencheront des actions, dans notre cas, la simulation des boutons de la boite.

```
<map name="map_default">
        <area shape="circle" coords="188,300,30" href="Page_2_Photo_FR.php" alt="pix">
</map>
```

Figure 5 : creation d'image map

On y trouve aussi la liste des documents créés. Cette page est réalisée en PHP avec un script listant les fichiers présents dans */home-www/ima4/jherin12/images*. L'utilisateur peut alors ouvrir ou supprimer la photo qu'il souhaite. Aucune gestion de droit n'a été implémentée donc tout le monde peut visionner et supprimer les fichiers.

2.2 Problèmes rencontrés

Un des premiers problème rencontré concerne le choix de la librairie graphique à utiliser pour gérer facilement l'ensemble de notre programme. Sur conseil de nos professeurs nous avons opté pour la librairie SDL. Nous avons cependant dû prendre en main à partir de zéro cette nouvelle façon d'utiliser le langage C. De plus, la bibliothèque est très riche et bien que les aides extérieurs (tutoriaux etc.) soient nombreux cela fut pour nous une difficulté.

Nous avons mis très longtemps à trouver une solution convenable et facile d'utilisation pour la gestion du flux vidéo. Une première solution était d'utiliser l'utilitaire vu en TP de réseau : **Gstreamer**. Même si la prise en main de cet outil aurait pris du temps, il semblait être la meilleure solution pendant les 8 premières semaines de projet. Cependant, afficher le flux vidéo comme nous le voulions c'est-à-dire avec une superposition d'image dans une fenêtre SDL, s'est révélé bien plus compliqué que prévu. Nous avons donc opté pour l'utilisation des fonctions native de **Video4Linux** qui est intégré au noyau linux.

Lors de nos essais d'affichage vidéo dans la fenêtre SDL, nous nous sommes rendus compte que la BeagleBone ne pouvaient traiter plus de 10 fps même dans les meilleures conditions. Après de très nombreux essais, nous avons dû faire un compromis sur taille de vidéo et le nombre d'image par seconde. Les essais comprenaient des modifications sur la taille de la vidéo, le format de pixel, la taille de couleur, le ratio. Dans le cas d'une trop haute résolution ou de fps trop élevés, la board ne répondait plus du tout. On peut voir à la ligne 333 le format par défaut, qui convient (*Figure 6)*.

```
int                 ifc, ninput = -1, width = 640, height = 400, gray = 0;
```

Figure 6 : Format convenant à l'affichage de la webcam sur l'écran

Ce problème est vraisemblablement dû au manque de puissance du CPU de la BeagleBone pour traiter des images HD (720p) en 24 fps.

Probablement pour la même raison, l'enregistrement et l'affichage simultanés d'une vidéo (et du son) ne fonctionnent pas. Nous avons effectué plusieurs tests à des résolutions minimes mais rien n'y fait.

2.3 Améliorations possibles

Voici quelques améliorations possibles de notre projet, en ce qui concerne la partie software. Dans un premier temps, le réel problème se situe au niveau de la prise vidéo. Il est nécessaire de faire fonctionner parfaitement la webcam c'est à dire :

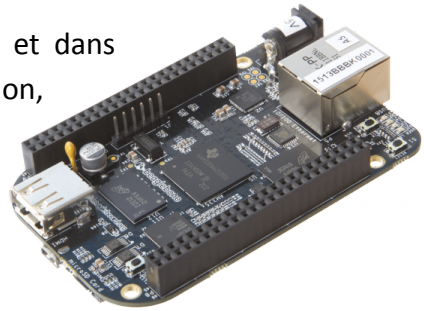
- Améliorer le temps de prise d'une photo
- Pouvoir prendre une vidéo et la visualiser
- Avoir une qualité d'image suffisante (720p et 24fps)

Ensuite, on peut évoquer la majeure partie des améliorations concerne le site. Il est nécessaire de soit transférer cette fonction vers un **Content Managment System** déjà existant (Wiki, blog ou autre), soit d'améliorer le site existant. On suggère dans ce deuxième cas de créer un système d'authentification, une réelle galerie et la possibilité de personnaliser chaque projet indépendamment. L'idée pour reconnaître qu'une photo appartient à un projet ou un porteur de projet particulier était d'utiliser des badges NFC. Ainsi en utilisant un badge spécifique, on indique à la boite quel projet on souhaite documenter.

3 Partie Hardware

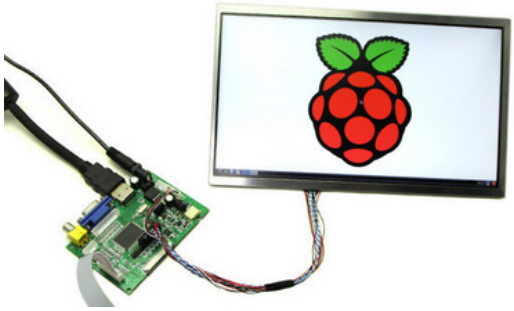
3.1 Matériel utilisé

3.1.1 La carte programmable



Afin de pouvoir gérer correctement un écran et une webcam, et dans l'optique de travailler sur un système performant pour notre utilisation, nous avons initialement pensé à une Raspberry Pi. N'étant pas disponible, il nous a été proposé une carte BeagleBone très semblable à la carte souhaitée.

3.1.2 L'écran



Nous avons souhaité utiliser un écran non tactile pour deux raisons :

La première pour limiter la consommation d'énergie et la deuxième pour rester dans l'optique d'une boite très simpliste où l'on utilise seulement deux gros boutons. En effet, l'aspect tactile d'un écran pourrait facilement repousser les personnes "allergiques" aux nouvelles technologies.

L'écran est un écran LCD 11" destiné aux cartes de type Raspberry Pi ou PCduino.

3.1.3 Les boutons

Afin d'obtenir une boite ludique et attractive, il nous avait initialement été suggéré d'utiliser de gros boutons de type champignon. Finalement, nous avons opté pour des boutons plus sobres, plus petits mais plus intuitifs.

3.1.4 La webcam

Le choix de la webcam est un des arguments principaux quant au choix de concevoir une boite entièrement plutôt que d'utiliser tout simplement une tablette tactile déjà existante et d'y implémenter notre propre application. En effet, nous pouvons utiliser la webcam de notre choix, possédant la résolution de notre choix. La boite est donc très flexible sur ce choix. Nous avons alors choisi une webcam Logitech C270 capable d'enregistrer des vidéos dans une résolution de 720p.

3.2 Travail réalisé

Avant de pouvoir recevoir le matériel, nous avons pu commencer à travailler sur la partie software sur une machine de l'école puisque le système sous lequel la boite fonctionne est Debian.

3.2.1 Les boutons

Une fois le matériel reçu, nous avons connecté les deux boutons principaux à deux des ports GPIO de la Beagle Bone *(Figure 7)*. Une résistance de pull-up de 10 kΩ a été choisie afin de limiter le courant arrivant sur les GPIOs de la carte.

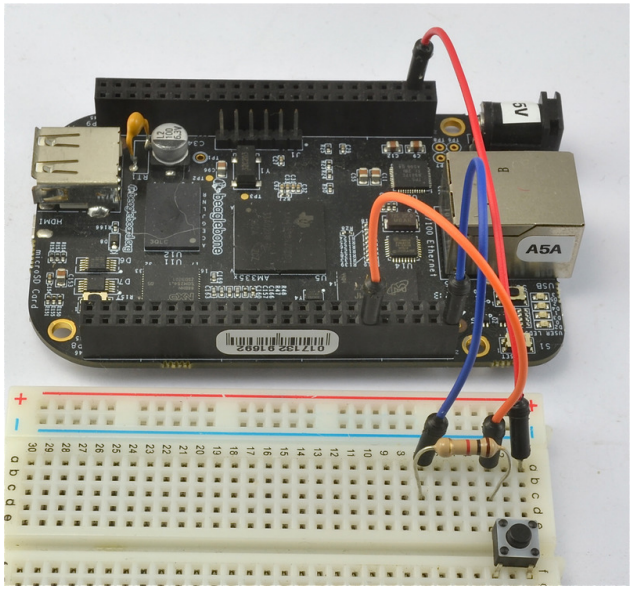


Figure 7 : Connexion d'un bouton sur la BeagleBone

Nous pouvons récupérer la valeur de l'état des boutons (1 ou 0) directement dans des fichiers sur la Beagle Bone *(figure 8)*. La lecture de ces fichiers est détaillée dans la partie Software de ce rapport.

Figure 8 : Récupération de l'état des boutons

Nous avons décidé d'ajouter deux boutons supplémentaires *(Figure 9)*. Le premier bouton permet de changer la langue de l'interface (français ou anglais). En effet, le campus Lille 1 compte aussi des personnes non francophones et/ou préférant l'anglais. Le Fabricarium étant ouvert à tous, il nous a semblé appréciable de pouvoir changer la langue de l'interface.

Le second bouton permet éteindre la boite. Le seul moyen aurait été de déconnecter le câble USB de la BeagleBone, ce qui pourrait être néfaste pour le matériel. Cela permet aussi de quitter le programme proprement.

Ces deux fonctions supplémentaires sont illustrées en **Annexe 1** et **Annexe 2**.

3.2.2 Les premiers prototypes

Nous avons organisé deux réunions avec des utilisateurs du Fabricarium et Rodolphe Astori afin de réfléchir à l'aspect de la boite. Afin de donner un aperçu lors de la première réunion, nous avons élaboré un premier prototype en carton à l'échelle *(Figure 9)*. De ces réunions en sont ressorties des idées de prototype en plexiglas afin de donner un aspect moderne.

Figure 9 : Premier prototype en carton

Pour des raisons pratiques et en attendant un prototype correct, nous avons utilisé une boite à chaussures (*Figure 10)* afin d'y installer tout le matériel et réaliser les premiers tests software. Cette boite nous aura aussi permis de la faire tester par plusieurs camarades ainsi que des utilisateurs du Fabricarium au niveau interface homme machine.

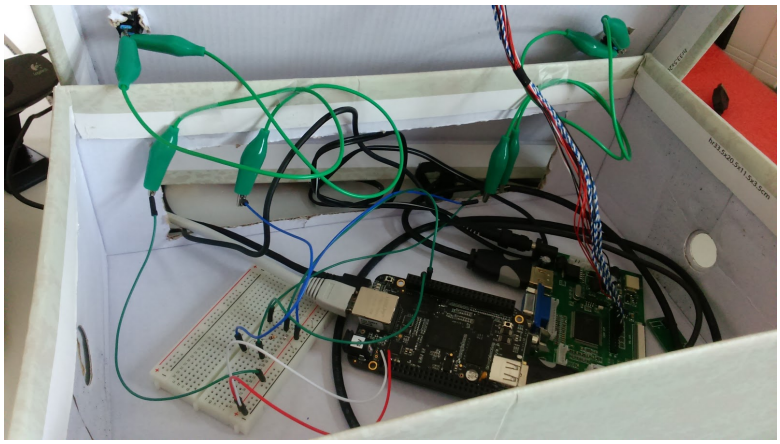


Figure 10 : Deuxième prototype avec une boite à chaussures

Avec l'aide de Mme Pichonat, nous avons appris à utiliser la découpeuse LASER du Fabricarium afin de réaliser un premier prototype correct en carton dur *(Figure 11)*. Ce prototype nous a permis de nous rendre compte des bonnes dimensions à choisir et de la crédibilité de la position de chacun des éléments. Nous ne sommes pas sans noter l'ironie d'utiliser le Fabricarium pour réaliser notre boite qui servira justement à cet espace.

Nous avons dessiné les pièces séparément à l'aide d'Inkscape, un logiciel de dessin vectoriel gratuit. Il a ensuite fallu découper chaque pièce séparément, avec la bonne puissance de laser.

Figure 11 : Troisième prototype en carton dur réalisé à l'aide de la découpeuse LASER

Pour finir, nous avons utilisé du bois aggloméré, réduit légèrement l'épaisseur de la boite et gravé le logo du Fabricarium à l'arrière de la boite et au niveau des boutons principaux *(Figure 12)*.

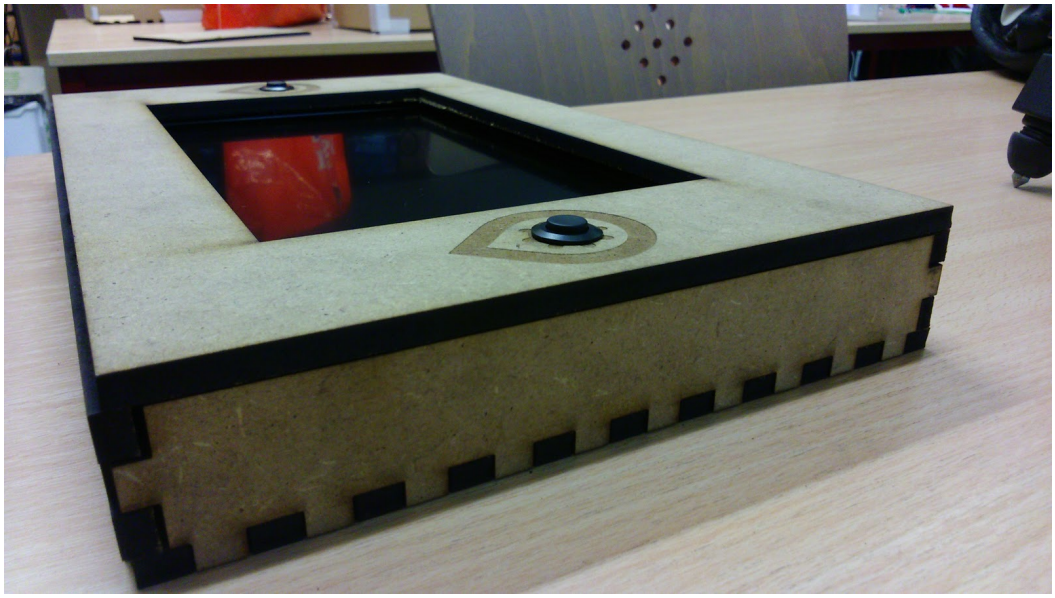


Figure 12 : Quatrième prototype en bois aggloméré réalisé à l'aide de la découpeuse LASER

La version finale *(Figure 13)* comprend deux boutons supplémentaires (décrits plus hauts) ainsi qu'une plaque de plexiglas protégeant l'écran de rayures et salissures.

Figure 13 : Version finale de la boite

3.3 Problèmes rencontrés

Au niveau de l'écran, les premiers tests ont révélé des extinctions et allumages successifs. Cela venait en réalité de l'alimentation qui délivrait seulement quelques milliampères. La solution était donc d'alimenter l'écran avec un courant de 2 ampères.

Pour l'alimentation, le côté portatif de notre projet n'est pas vérifié puisque pour l'instant la carte vidéo nécessite une alimentation murale. En effet, très peu de batteries portatives présentes aujourd'hui sur le marché permettent une autonomie convenable de notre boite.

3.4 Améliorations possibles

La prise en main de la boite pourra être facilitée par l'ajout de poignées confortables telles que sur la *figure 14*.

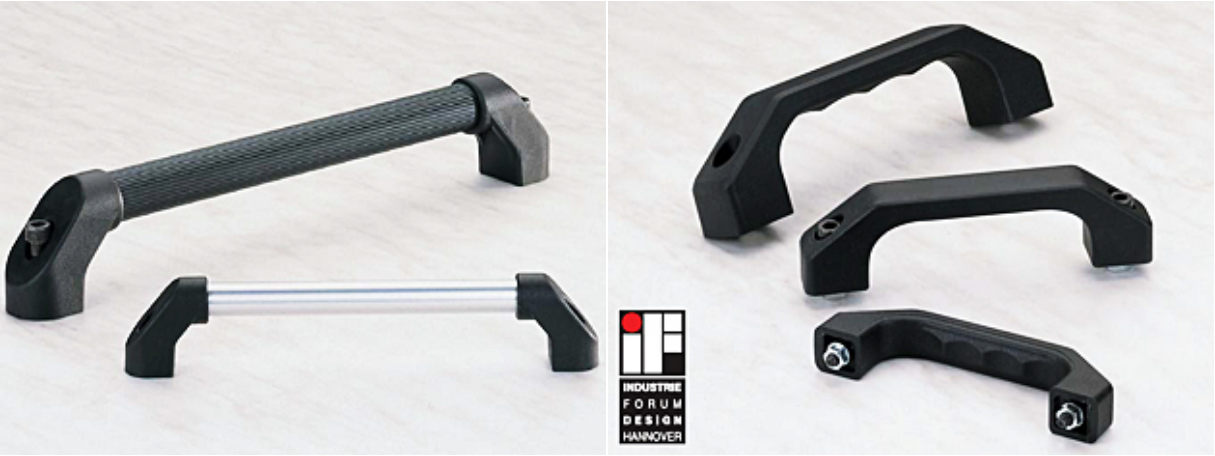


Figure 14 : Exemples de poignées confortables

Côté esthétique, nous avons pensé à l'utilisation de plexiglas, qui est très moderne et permettant aux utilisateurs de se rendre compte du matériel présent au delà d'un écran et de quatre boutons. De plus, les possibilités de plier le plexiglas permettent des formes plus ergonomiques.

La boite peut aussi être plus légère et plus fine telles les tablettes tactiles que l'on peut rencontrer sur le marché actuel. Les cartes actuelles présentes dans la boite ne permettent malheureusement pas cette amélioration.

Bilan

Ce projet nous a confronté à la difficulté de définir nous-mêmes le cahier des charges en collaboration avec le client, avec des modifications a posteriori et de trouver les solutions les plus crédibles à la réalisation du projet.

Ce projet fut plein de surprises pour nous. Nous ne nous attendions pas à rencontrer de telles difficultés et avons passé beaucoup de temps sur des problèmes inattendus. Ce fut évidemment bénéfique car nous avons beaucoup appris, tant dans la gestion de projet, qu'en C avec l'utilisation de différentes librairies et en conception mécanique avec la création de la boite et l'utilisation d'une imprimante LASER.

Cependant, l'impression de ne pas avancer et de bloquer sur des problèmes simples a aussi été un grand défi que nous avons dû surmonter. Au final, le cahier des charges est en grande partie respecté mais nous ne sommes pas pleinement satisfaits du prototype que nous livrons.

Annexes

- Choix de la langue
- Extinction de la boite
- Choix du mode (Vidéo / Photo)
- Retour webcam avant prise de photo
- Retour de la photo prise
- Page finale précisant à l'utilisateur où trouver ses photos

ANNEXE 1

Choix de la langue

ANNEXE 2

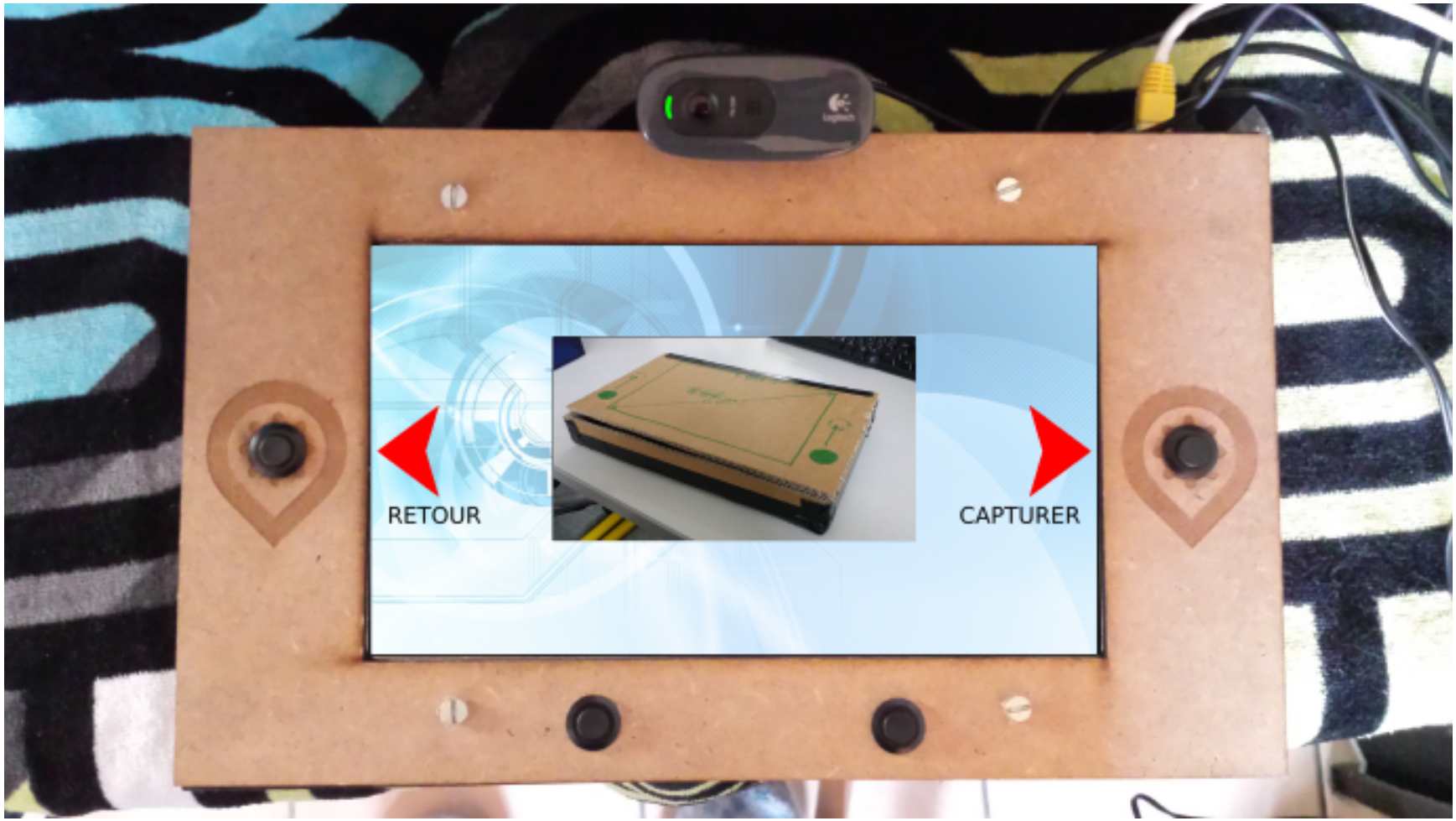
Extinction de la boite

ANNEXE 3

Choix du mode (Vidéo / Photo)

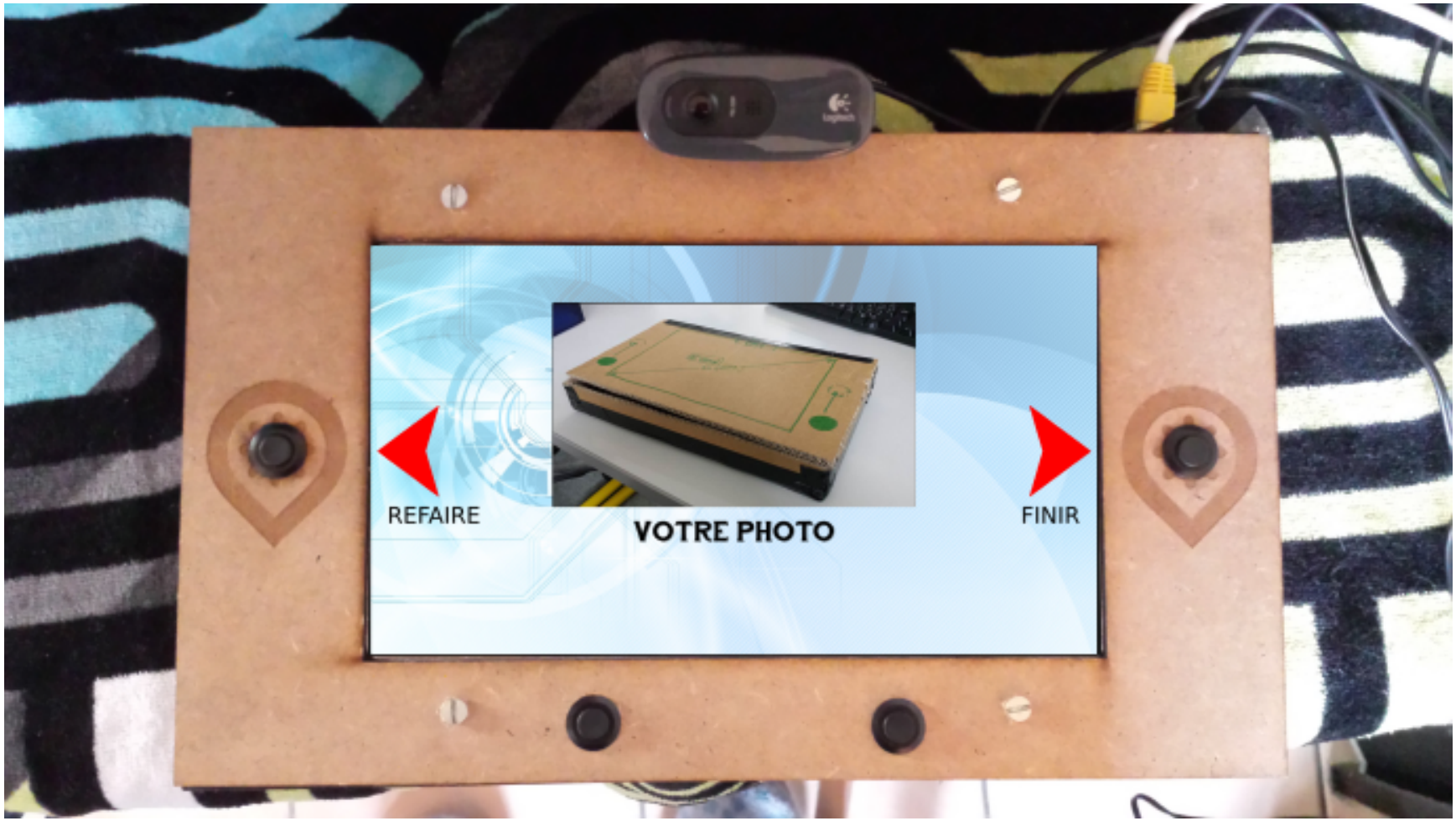
ANNEXE 4

Retour webcam avant prise de photo



ANNEXE 5

Retour de la photo prise



ANNEXE 6

Page finale précisant à l'utilisateur où trouver ses photos