

|  |
| --- |
| Projet et bureau d’étude  ScanBook |

Rapport d’avancement n°2

Rapport bimensuel du 06/03/2017 au 28/03/2017

Thomas Herpoel – Wesley Estievenart – Michael Manzella

Table des matières

[Aspects techniques 3](#_Toc478490603)

[Problèmes techniques 6](#_Toc478490604)

[Problèmes d’ordre organisationnel/pratique 6](#_Toc478490605)

[Liste des décisions/orientations prises avec l’approbation du client 6](#_Toc478490606)

[Résultat test USART 6](#_Toc478490607)

# Aspects techniques

WP\_01 : Test de la caméra

Un premier test a été réalisé avec 2 modèles de caméra pour Raspberry Pi : le modèle 1.3 et le modèle 2.1. Ces deux caméras ont fonctionné avec la configuration officielle du Raspberry Pi. Des prises de vue préliminaires indiquent que la résolution du capteur de la caméra 2.1 serait suffisante, et peut-être même celle du capteur de la caméra 1.3. Ces résultats ont besoin d’être confirmés en situation réelle, dans le système final.

WP\_02 : Choix de la bibliothèque de reconnaissance d’image et de l’algorithme (OCR)

La bibliothèque de reconnaissance d’image utilisée sera OpenCV dans son implémentation en C++. Le système de reconnaissance de caractère n’est pas encore défini.

WP\_03 : Programmation traitement des images

Les tests effectués ont permis de prendre en main le développement d’application OpenCV en C++ et permettent d’effectuer la reconnaissance des contours d’un livre pour prouver le bon fonctionnement de la librairie. Voir figure 1.

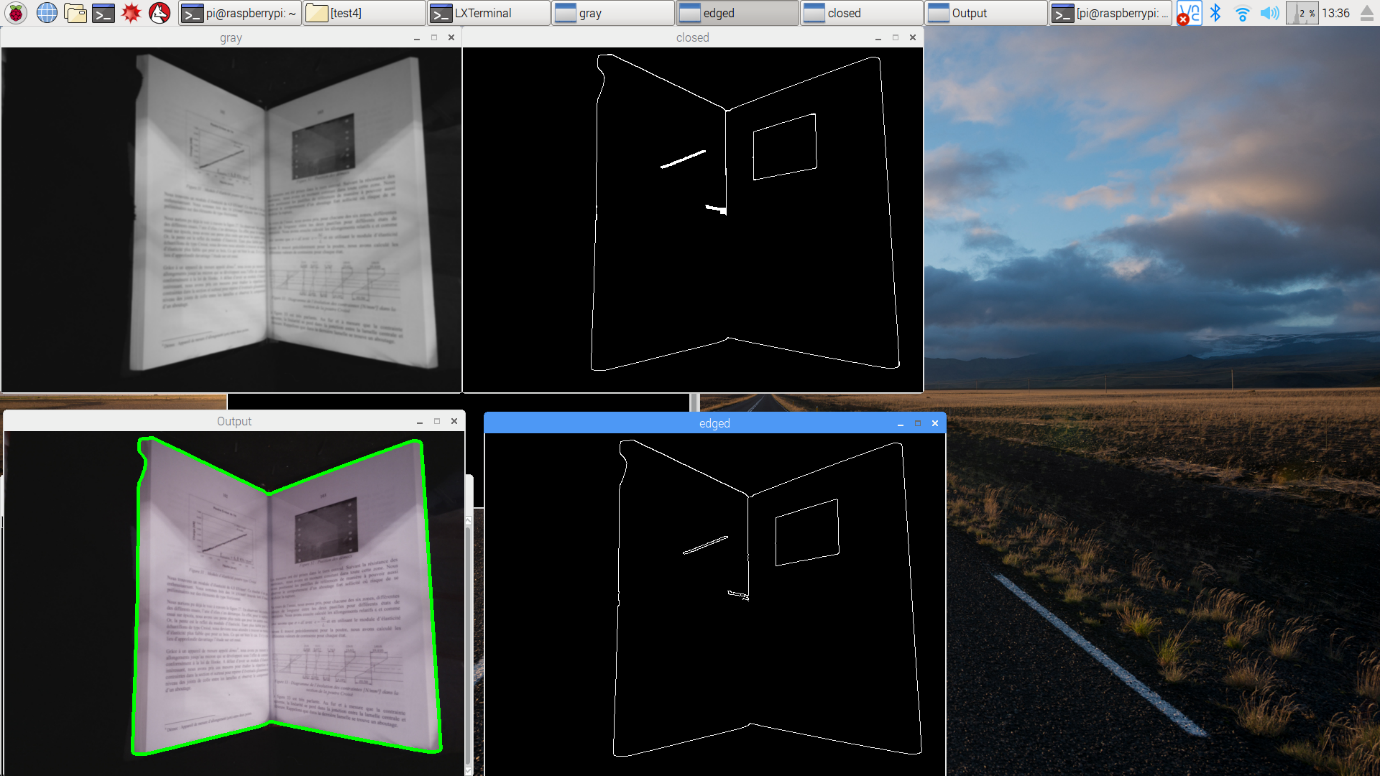


Figure 1 : tests OpenCV

WP\_04 : Modification de la structure actuelle

La structure est complètement différente. Le design sur logiciel de CAD (Autodesk 360 Fusion) couplé à l’utilisation de l’imprimante 3D du Cerisic ont permis de développer une structure modulaire et robuste. Le rendu du premier jet de cette conception CAD est montré figure 2. La figure 3 présente l’avancement de la réalisation de la structure lors du premier rapport d’avancement et la figure 4 montre la structure telle qu’elle est actuellement.

|  |  |
| --- | --- |
| C:\Users\michael\Desktop\Annee 2016-2017\Projet Master\renduHD.png  Figure : Rendu de la structure sur le logiciel Fusion 360 | C:\Users\Thomas\AppData\Local\Microsoft\Windows\INetCache\Content.Word\IMG_20170306_171104.jpg  Figure : Début du montage |

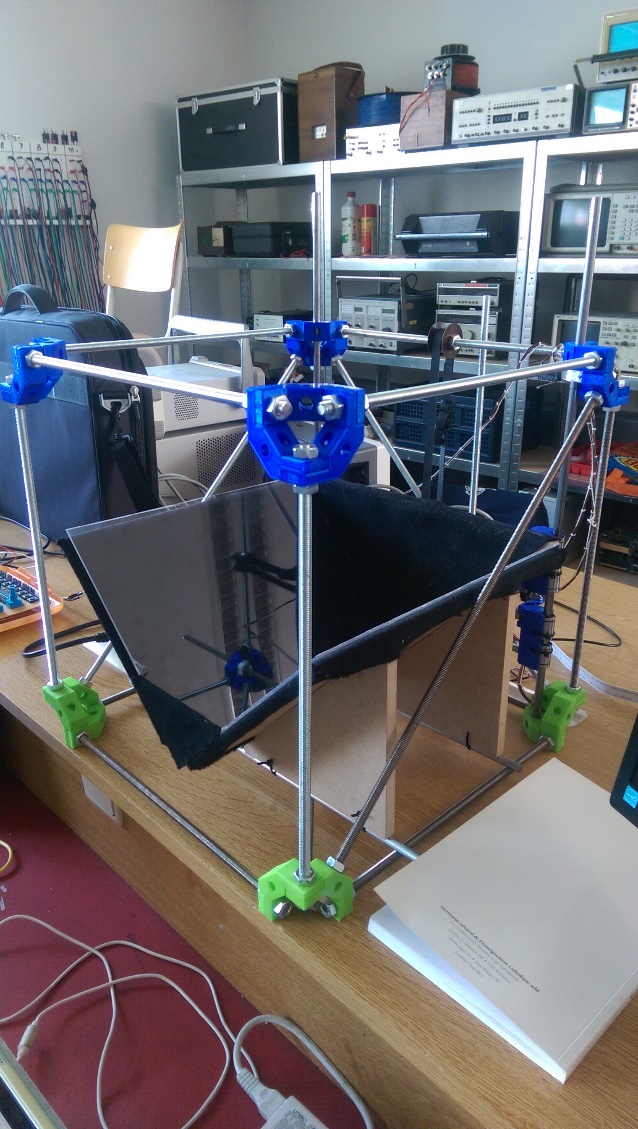


Figure 4 : structure actuelle

Certaines parties, telles que le support du livre ou les fixations de la partie mobile, n’ont pas été correctement dessinées et modélisées sur ordinateurs par manque de temps. Cependant, la créativité de l’équipe de projet couplée au faible prix des attaches de type Colson® permet de fournir un prototype fonctionnel. La fixation des caméras doit encore être réalisée.

|  |  |
| --- | --- |
| Figure 5: face arrière | Figure 6 : fixation moteur |
| Figure 6: fin de course haut | Figure 7 : fin de course bas |

WP\_05 : Conception d’un module de switch caméra (PCB)

Afin de connecter 2 caméras sur un seul Raspberry, un circuit a été dessiné basé sur la puce FSA642 de fairchild pour switcher les signaux MIPI des caméras et une puce mux/demux 74CBTLV3257D pour switcher les signaux I2C de configuration des caméras. Ce circuit est montré en figure 9, et le cuivre des 2 couches du PCB dessiné avec Eagle est montré en figure 10 et 11.

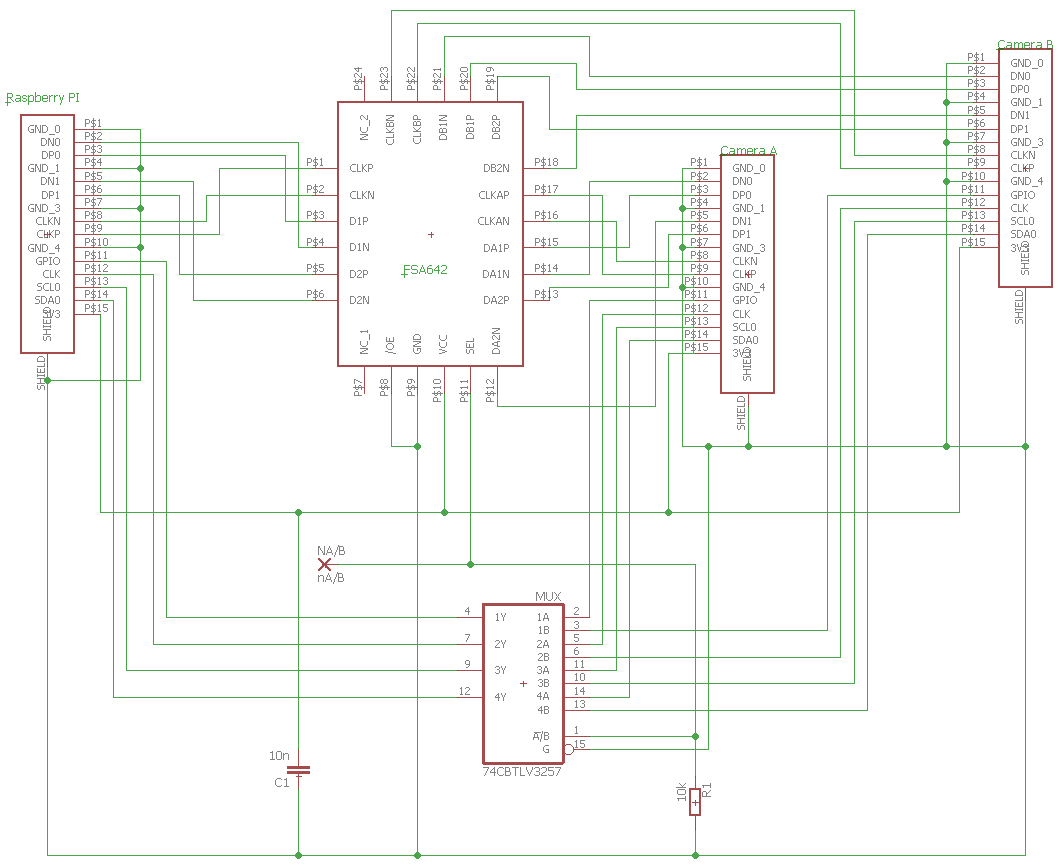


Figure 9 : Circuit switch

|  |  |
| --- | --- |
| C:\Users\Thomas\AppData\Local\Microsoft\Windows\INetCache\Content.Word\pcb_top.png  Figure 10: PCB top layer | C:\Users\Thomas\AppData\Local\Microsoft\Windows\INetCache\Content.Word\pcb_bottom.png  Figure 11 : PCB bottom layer |

WP\_06 : Circuit de protection (moteur pas à pas)

* Choix de la configuration hardware (broche du module de contrôle du moteur "enable" mise à la masse de sorte à maintenir le moteur alimenté)

WP\_08 : Programmation commande moteur

* Modification de la programmation du microcontrôleur de l’année précédente
* Commande de test fonctionnelle (PC vers microcontrôleur, PC vers Raspberry et Raspberry vers microcontrôleur)
* Définition de la séquence de fonctionnement

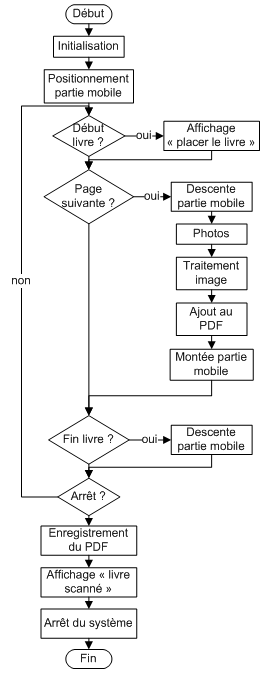


Figure 12 : Séquence programme Scanbook

WP\_09 : Interface homme-machine

* En attente de la réception de l’écran TFT 320x480 pixels tactile, utilisation d’un écran LCD 2 lignes 16 caractères
* Réception de l’écran tactile et configuration

WP\_10 : Système d’alimentation

* Réception du bloc d’alimentation 12V 2A

WP\_11 : Programmation Raspberry Pi (configuration, communication, …)

* Suite et fin de la configuration de l’USART
* Tests commandes moteurs (simulation avec le programme Docklight et test sur la structure physique)
* Installation complète de la bibliothèque OpenCV pour le Raspberry Pi

# Problèmes techniques

WP\_04 : Modification de la structure actuelle

* Recherche d’une solution efficace pour la partie mobile plus difficile que prévue

WP\_05 : Conception d’un module switch caméra (PCB)

* Mauvais choix de composant. Le circuit SN74HCT157 est un simple multiplexeur. Or dans notre cas, il faut un circuit qui permette le passage des signaux dans les deux directions. Une autre puce ayant le même pinout dans le même package et offrant cette possibilité a été trouvée et commandée : le circuit SN74CBTLV3257D.
* Le PCB soudé ne fonctionne pas. La soudure de la puce en package QFN est peut-être responsable. Une solution à ce problème n’est pas encore trouvée. Une autre cause pourrait être la haute fréquence des signaux MIPI CSI des caméras. L’impédance des lignes différentielles sur le PCB n’est peut-être pas adéquate. La possibilité d’une erreur de conception du circuit est aussi à envisager.

WP\_11 : Programmation Raspberry Pi

* Problème lors de l’activation et la configuration de l’USART sur le Raspberry Pi 3 🡪 Résolu
* Problème lors de l’installation et la compilation de la bibliothèque d’OpenCv sur le Raspberry Pi B+ 🡪 Résolu

# Problèmes d’ordre organisationnel/pratique

Commande de l’échantillon pour la puce du switch de caméra confirmée, mais pas de nouvelles depuis (10-02-2017). 🡪 Résolu : Nouvelle commande effectuée sur DigiKey et reçu

Commande de divers éléments venant de Chine à faire d’urgence. 🡪 Résolu : Réception de tous les composants

# Liste des décisions/orientations prises avec l’approbation du client

|  |  |  |
| --- | --- | --- |
| WP concerné ou nouveau WP | Description de la décision | Date de validation par le promoteur |
| WP\_01 | * Prise de vue caméra : passé de 1 à 2 caméras * Utilisation de caméra Pi (Résolution de 5M et 8M Pixels) |  |
| WP\_04 | * Structure du caisson mobile * Modélisation des pièces pour la structure par Imprimante 3D |  |
| WP\_08 | * Gestion commande moteur avec Raspberry Pi et ATmega |  |
| WP\_11 | Rendre le ScanBook indépendant d’un PC avec l’utilisation d’un Raspberry Pi |  |

# Résultat test USART

La communication entre le Raspberry Pi 3 et le microcontrolleur ATmega88 se fait via Usart. Il a fallu configurer l’USART sur le Raspberry dans un premier temps avant de pouvoir l’utiliser. Une fois cette étape franchie, les différentes commandes pour contrôler le Scanbook on été elaborées.

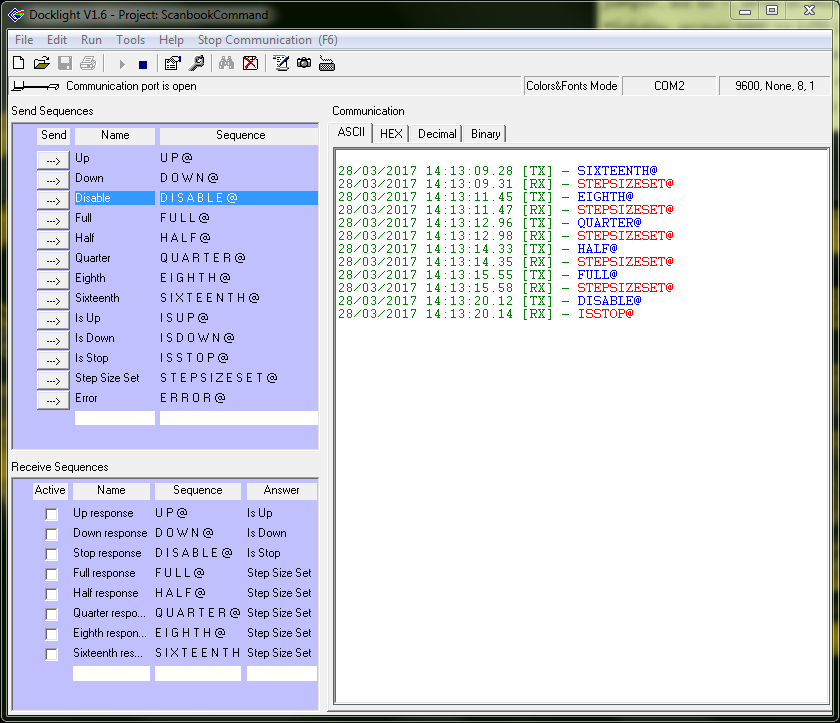


Figure 13 : Commandes réglage résolution

Sur la Figure 13, on peut voir l’envoi de commande du pc (en bleu) avec le Docklight vers le microcontrôleur. Pour ce qui est du réglage de la résolution, le microcontrôleur répondra (en rouge) par la même chaine de caractère (cela dans le but de réduire et simplifier les commandes). On peut également voir une commande permettant l’arrêt du moteur si on le souhaite (pour l’arrêter pendant un déplacement par exemple, entre les deux fin de course) (voir Figure 14). On peut voir sur le panneau de gauche les différentes commandes configurées dans Docklight afin de réaliser tous les tests sur la communication USART. On peut aussi voir dans le bas, des réponses automatiques utilisées lors de test plus poussés sur les programmes pour simuler au mieux le fonctionnement réel de la communication entre le Raspberry et le microcontrôleur.

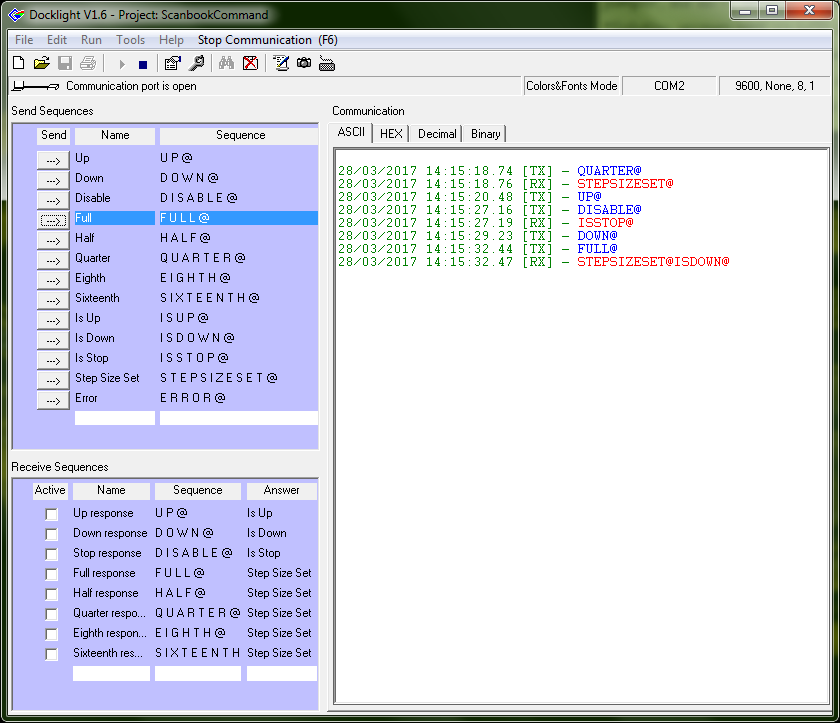


Figure 14 : Arrêt moteur et changement résolution pendant fonctionnement

On peut voir sur la Figure 14, un arrêt du moteur pendant la montée et un changement de la résolution pendant la descente.

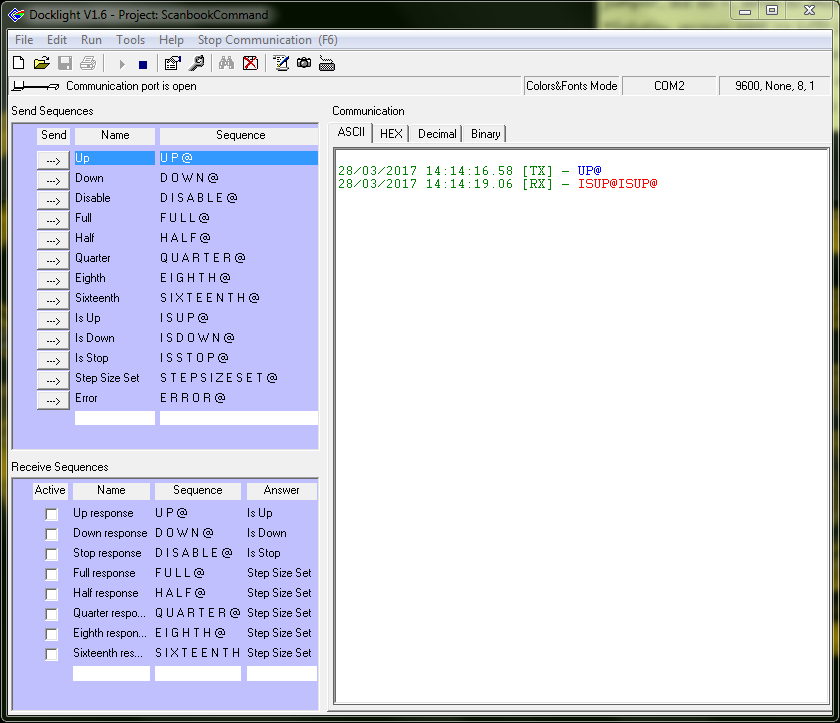


Figure 15 : Commande montée Scanbook

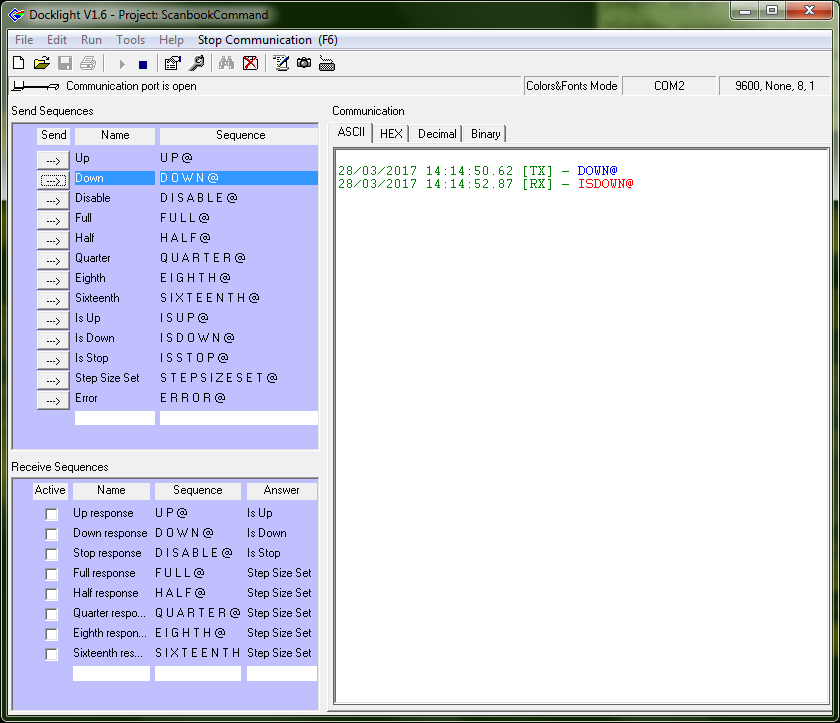


Figure 16 : Commande descente Scanbook

On peut voir sur les Figure 15 et Figure 16, respectivement les commandes pour la montée et la descente des vitres envoyée (en bleu) vers le microcontrôleur. On voit la réponse du microcontrôleur (en rouge) lorsque la partie mobile actionne les fin de course haut et bas. Pour le fin de course haut, on peut remarquer que l’envoi s’est effectué deux fois. Cela est très certainement dû à un rebond du fin de course.

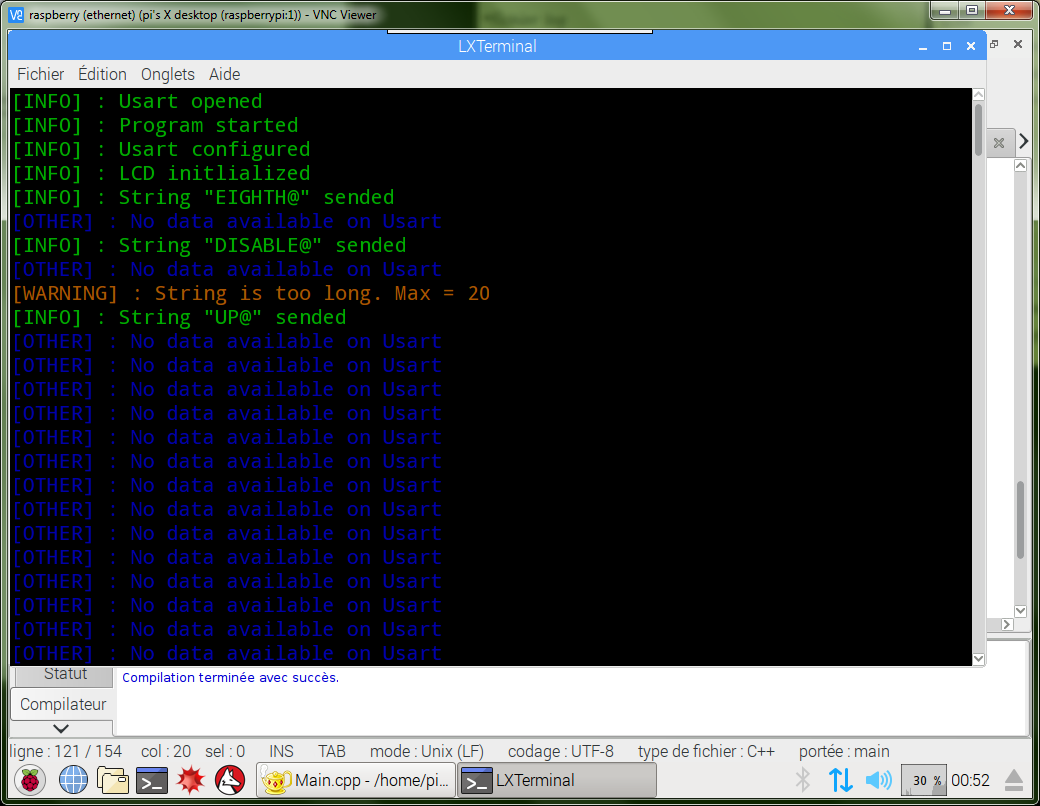


Figure 17 : Console Raspberry communication USART

Sur la Figure 17, on peut observer la console du Raspberry une fois le programme du Scanbook lancé. Un affichage couleur a été implémenté afin de visualiser plus clairement et rapidement les informations voulues. Il y a plusieurs type d’informations affichées et une couleur correspondant à chacune d’entre-elles. Les erreurs seront en rouges, les attentions en orange, les autres informations en vert ou en bleu selon leurs utilisations. On peut donc suivre le déroulement du programme et rapidement savoir où il en est. Sur cette capture d’écran, on peut voir une série d’initialisation comme l’USART ou le LCD (prévu en attente de la réception de l’écran tactile). On peut ensuite voir l’envoi en USART de deux commande, une pour régler la résolution et l’autre pour s’assurer que le moteur soi à l’arrêt. Vient ensuite un test d’envoi d’une chaine trop longue, un message d’avertissement apparait en orange pour le signaler. Enfin, on voit que le Raspberry est en attente d’une réception USART.

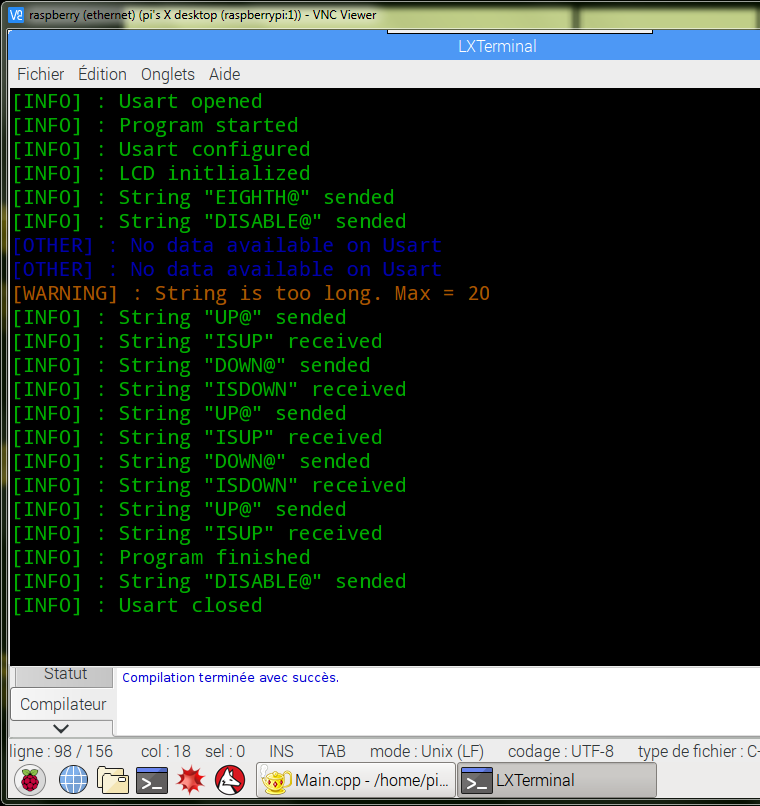


Figure 18 : Test commande Scanbook

Sur la Figure 18, on peut voir un programme de test sur le Raspberry. Celui-ci consistait à faire monter la partie mobile, une fois en haut la faire attendre un certain temps, la faire redescendre et enfin attendre. La séquence tournait en boucle afin de simuler le fonctionnement final de l’application et de tester les programme en fonctionnement continu avec la structure physique. Sur la Figure 19, on peut voir la simulation de la partie microcontrôleur avec Docklight. On voit les commandes envoyées par le Raspberry (en rouge) et celle répondue (en bleu). Le programme a donc été testé en simulation avant de le tester sur la structure physique, afin de s’assurer qu’aucune erreur n’était commise.

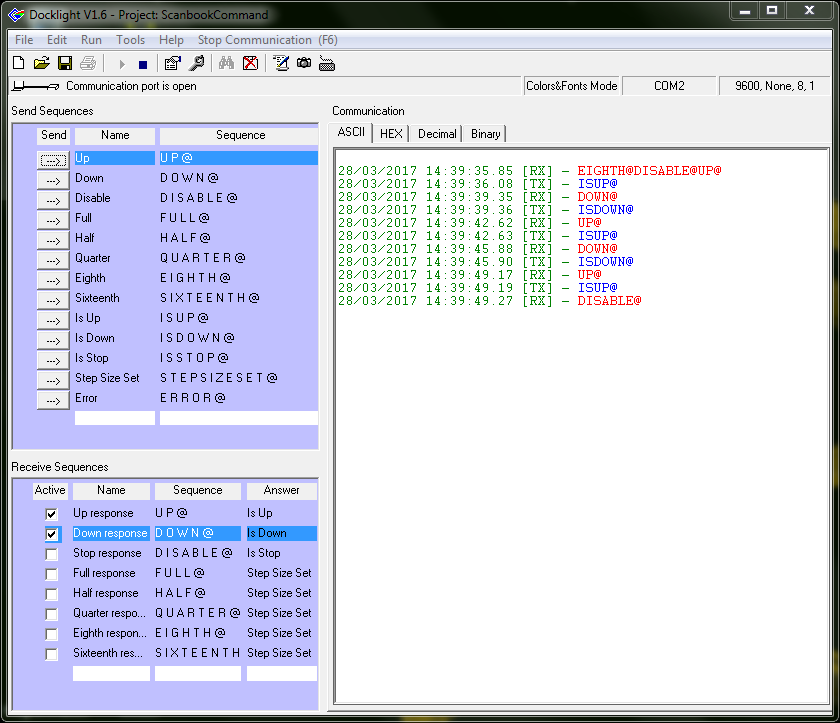


Figure 19 : Simulation microcontrolleur avec Docklight