

**Nano Ordinateurs  
Gestion de projet   
Robot Baliseur**

Mars 2018

Version 1.0

Wattin Jérôme  
Sam Mahaux

Fabio Cumbo

Timothée Simon

# Informations générales sur le document

### Contact

Pour toute question ou remarque concernant ce document, merci de contacter :

Wattin Jérôme - Etudiant

Téléphone : +32479886237

Mail : jerome.wattin@std.heh.be

HEH Campus Technique

8A Avenue Maistriau,

7000 Mons

### Confidentialité

Ce document contient des informations confidentielles et exclusives de la Haute École en Hainaut (HEH). Le service informatique ne peut divulguer les informations confidentielles contenues dans ce document à un tiers sans le consentement écrit de la HEH, hormis aux employés, enseignants ou directeurs qui ont besoin de connaître son contenu à des fins d'évaluation du document. Le service informatique se doit d'informer ces personnes de la nature confidentielle de ce document et d'obtenir leur accord pour préserver sa confidentialité.

### Termes et conditions

La HEH n'assume aucune responsabilité pour les erreurs ou omissions dans le contenu de ce document ou de tout document de tiers référencé ou associé, y compris, mais sans s'y limiter, les erreurs typographiques, les inexactitudes ou les informations périmées. Ce document et tous les renseignements qui s'y trouvent sont fournis «tels quels» sans aucune garantie, expresse ou implicite.

### Informations sur le document

|  |  |
| --- | --- |
| Nom du document | RobotBaliseurQ2.Docx |
| Version | Version 1.00 |
| Niveau de confidentialité | Utilisation interne uniquement |
| Auteur du document | Wattin Jérôme |
| Contributeur(s) |  |
| Révisé par |  |
| Approuvé par |  |

### Versions

|  |  |  |  |
| --- | --- | --- | --- |
| version | Date de parution | Modification réalisée par | Modification(s) apportée(s) |
| 1.00 | 26/03/2018 | Wattin.J | Création du document |
|  |  |  |  |
|  |  |  |  |

### Documents connexes et/ou de référence

|  |  |  |  |
| --- | --- | --- | --- |
| Nom du document | Description | Date | version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### Table des matières

[1. Informations générales sur le document 3](#_Toc514415248)

[Contact 3](#_Toc514415249)

[Confidentialité 3](#_Toc514415250)

[Termes et conditions 3](#_Toc514415251)

[Informations sur le document 4](#_Toc514415252)

[Versions 4](#_Toc514415253)

[Documents connexes et/ou de référence 4](#_Toc514415254)

[Table des matières 5](#_Toc514415255)

[2. Introduction et présentation générale du projet 6](#_Toc514415256)

[3. Matériel utilisé 6](#_Toc514415257)

[4. Présentation détaillée du capteur à ultrason 7](#_Toc514415258)

[5. Contrôle du robot 7](#_Toc514415259)

[6. Choix de l’IDE 9](#_Toc514415260)

[7. Algorithme de programmation 9](#_Toc514415261)

[7.1. Programmation du robot 9](#_Toc514415262)

[7.2. Développement du site web 9](#_Toc514415263)

[8. Problèmes rencontrés 10](#_Toc514415264)

[9. Améliorations 10](#_Toc514415265)

[10. Conclusions 10](#_Toc514415266)

# Introduction et présentation générale du projet

L’objectif principal de ce projet a été de programmer une voiture afin de lui permettre de réaliser une cartographie autonome de son environnement.

Nous avons été poussé à choisir le robot baliseur pour le côté original du projet et parce que nous avons estimé le sujet intéressant ainsi que le travail de recherche derrière qui pourraient nous apporter de nouvelles notions.

# Matériel utilisé

Pour réaliser cela, nous avons utilisé une base d’un robot de marque Initio composée d’un châssis, 4 roues et 2 moteurs pas à pas.

Nous avons également utilisé 2 capteurs à ultrason pour détecter les obstacles ainsi qu’un Raspberry pi 3B pour programmer le robot.

Afin de sécuriser au mieux son déplacement, nous avons fixé un capteur à ultrason à l’avant du robot et l’autre sur le côté gauche.

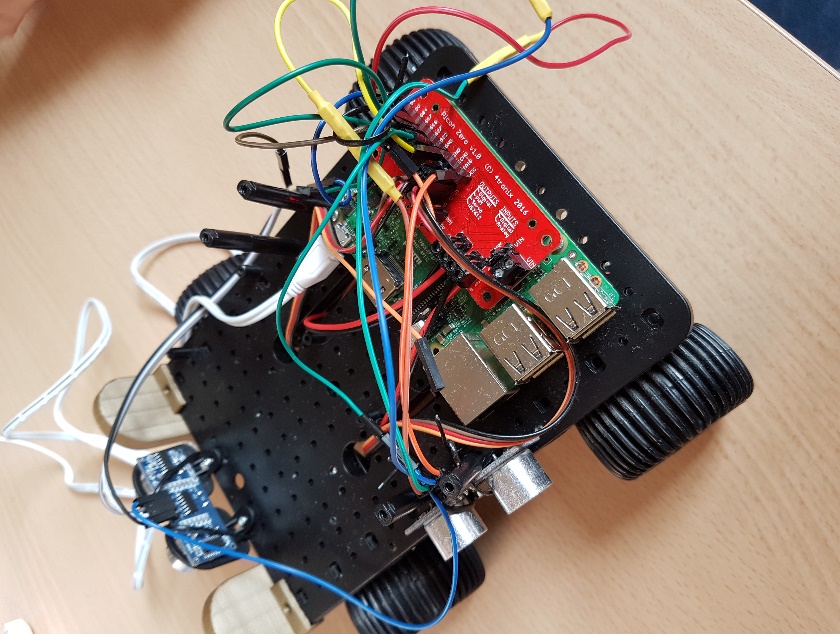


Photo du robot monté

# Diagramme de Gantt

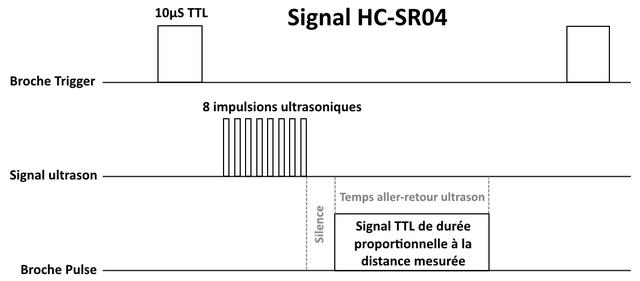
# Présentation détaillée du capteur à ultrason

**PRINCIPE PHYSIQUE, ELECTRONIQUE ET PROGRAMMATION DEMANDEE, PARTIE MICHIELS DONC IMPORTANT**

[**https://www.gotronic.fr/pj2-hc-sr04-utilisation-avec-picaxe-1343.pdf**](https://www.gotronic.fr/pj2-hc-sr04-utilisation-avec-picaxe-1343.pdf)

**https://www.carnetdumaker.net/articles/mesurer-une-distance-avec-un-capteur-ultrason-hc-sr04-et-une-carte-arduino-genuino/**

Le capteur à ultrason fonctionne sur base de la vitesse du son.



La prise d’une mesure se déroule comme suit :

1. On envoie une impulsion HIGH durant 10 µs sur la broche Trigger (pin nommé « Trig ») du capteur
2. Le capteur répond à l’impulsion, par l’envoie d’une série de 8 impulsions ultrasonique (40KHz), inaudible pour l’être humain.
3. A l’impact d’un objet, l’ultrason retourne en sens inverse. Cela signifie que lors d’un impact face à la source, l’onde est renvoyée en direction du capteur
4. Lors de la réception de l’écho, le capteur clôture.

Afin de déterminer la durée de l’aller-retour de l’onde, la broche ECHO du capteur (pin nommé « echo ») reste sur HIGH durant les étapes 3 et 4.

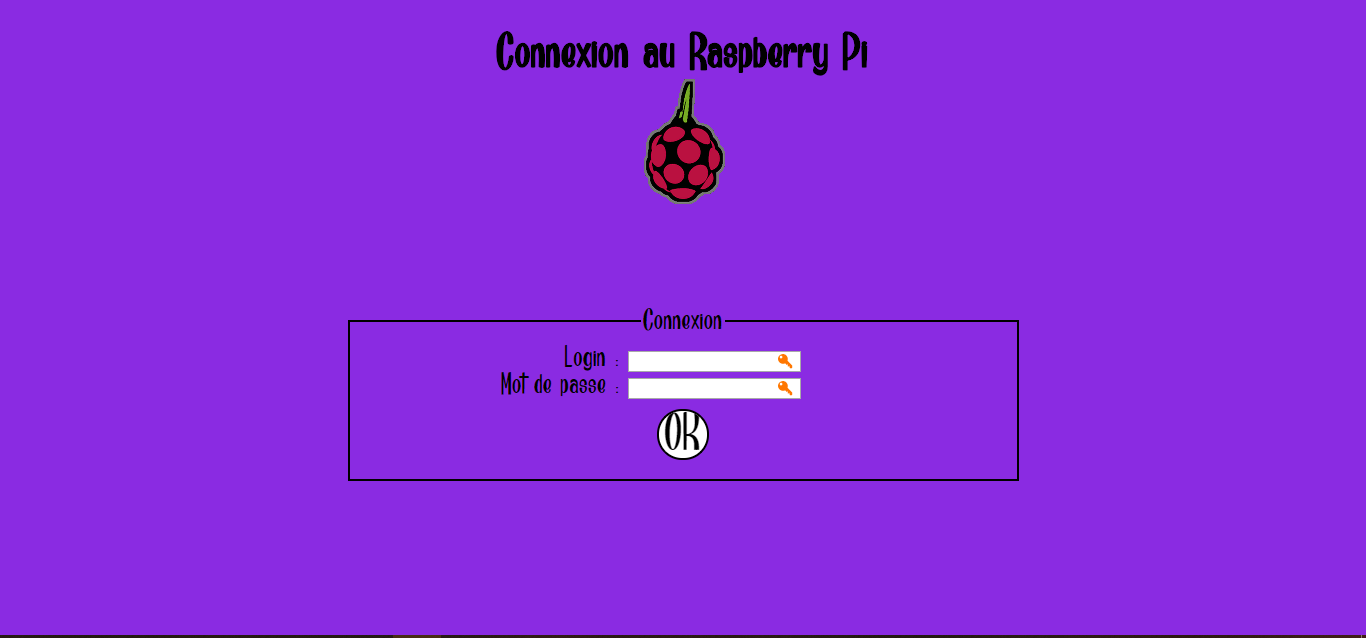
# Contrôle du robot

Pour tirer parti au mieux du Raspberry pi 3 et pour le côté « innovation » du projet, nous avons décidé de développer un site web. Cette solution nous a permis à ce que le contrôle du robot soit opérationnel avec pratiquement tout les supports.

L’objectif premier du site était d’obtenir une interface pour démarrer le robot afin de cartographier une pièce et l’afficher en retour.

Nous avons donc dû installer un serveur Apache 2 sur le Raspberry pi 3 permettant de répondre aux requêtes envoyées par l’utilisateur, et nous avons également installé MySQL et PHP sur celui-ci.

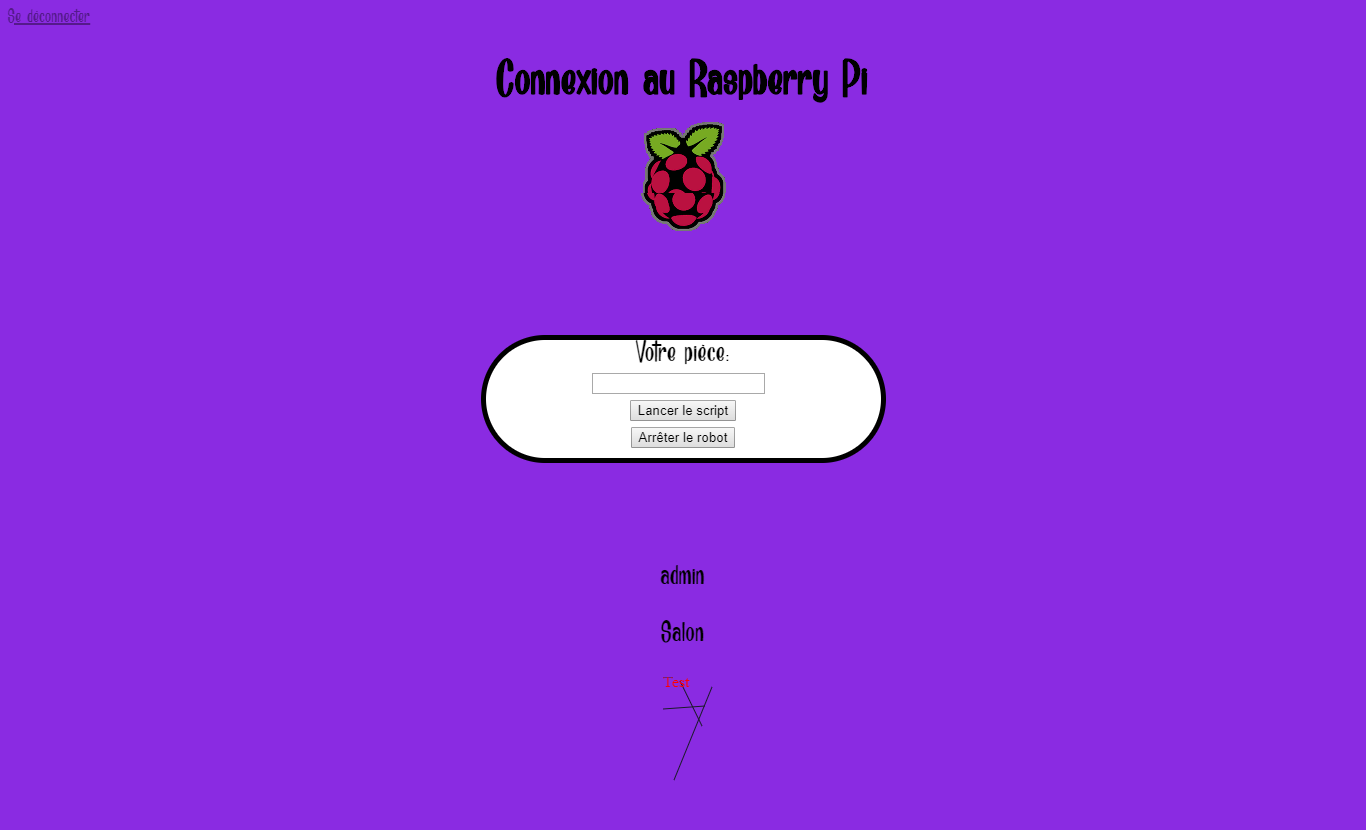
Un minimum de sécurité sur le site web a été implémenté afin d’éviter que n’importe qui puisse démarrer une cartographie. Une interface de connexion a donc été mise en œuvre. Pour stocker les comptes utilisateurs et se connecter, nous avons ajouté une table appropriée à notre base de données.



Interface de connexion du site web

Afin d’apporter un intérêt supplémentaire au site web, nous avons également créée une autre table dans la base de donnée pour stocker l’utilisateur, la pièce indiquée et sa cartographie.

Chaque utilisateur peut donc visualiser son historique de cartographie effectué.



Interface permettant de lancer le script pour cartographier et de visualiser ses cartographies déjà effectuées

# Choix de l’IDE

Notre choix d’IDE s’est porté sur Geany.  
En effet, celui-ci étant léger et équipé d’un interpréteur intégré, il inclut les fonctionnalités élémentaires pour la réalisation de notre projet. De plus il est multi-plateforme et supporte les langages que nous avions dû utiliser, c’est-à-dire Python, HTML, CSS et PHP.

Il répondait donc à nos exigences.

# Outils utilisés pour la gestion du projet

Afin de permettre une gestion de projet efficace, nous avons tous utilisé Git et nous avons aussi créée un dépôt sur Github.

Cela nous a principalement permis de synchroniser toutes nos modifications effectuées. Tout le monde a donc pu travailler en parallèle sur le projet ce qui a amélioré la gestion générale de celui-ci et nous avons également pu gérer les différentes versions du robot baliseur.

L’utilisation de Github pourra nous permettre en autre de partager notre projet à une plus large communauté.

Nous avons également utilisé Microsoft Project pour visualiser la répartition des tâches et obtenir le Diagramme de Gantt.

# Algorithme de programmation

## Programmation du robot

## Développement du site web

# Problèmes rencontrés

* Les premiers problèmes auxquels nous avons été confrontés ont été la fixation et le montage des 2 capteurs à ultrason sur le PiconZero.

En effet, étant donné que ce dernier ne possédait qu’une seule entrée TRIG et ECHO, nous avons du nous rabattre sur les port GPIO du Raspberry Pi 3 pour brancher et faire fonctionner les 2 capteurs à ultrason.  
Au niveau des fixations des capteurs, nous avons dû réutiliser celles du toit afin de pouvoir les poser latéralement. ??

* L’exécution de script Python par le site WEB et donc par le code PHP nous a particulièrement posé problème.  
  En effet, par défaut l’utilisateur créée par Apache (www-data) n’a pas les droits pour réaliser certaines actions, il a donc fallu éditer le fichier sudoers via la commande *sudo visudo* et donner les droits à l’utilisateur www-data sur les scripts visant à être exécutés .   
  Il a également fallu mettre l’utilisateur www-data comme propriétaire avec le groupe root sur les fichiers PHP et Python voulu.
* Nous avons eu des soucis pour permettre d’arrêter le script Python directement depuis le site web.   
  Effectivement étant donné que lorsque l’on démarré une cartographie, un script Python pour cartographier était appelé et tant que son exécution n’était pas terminé, la page web indiqué toujours un chargement puisque le code PHP n’avait également pas finis d’être exécuté.   
  Dès lors lorsque l’utilisateur souhaité stopper la cartographie, le code PHP permettant d’appeler un autre script Python pour arrêter le robot ne pouvait donc pas être exécuté tant que le navigateur chargé toujours la page.  
  Pour arrêter le robot depuis le site web il fallait donc cliquer sur la croix du navigateur et ensuite cliquer sur le bouton pour arrêter la cartographie.
* SVG…
* Correction de mouvements…

# Améliorations

* Au niveau du site web nous aurions pu améliorer la fonction permettant d’arrêter le robot depuis le site web avec l’utilisation de JavaScript et d’AJAX.

Nous aurions pu lancer le script python en AJAX (XMLHttpRequest) et l’arrêter avec du JavaScript (req.abort()) pour résoudre ce problème mais nous n’avons pas eu le temps pour effectuer cela.

* SVG

# Conclusions

# Annexe

Tous les codes