



## Livrable 3 - Robot Kinocito

### Design III

présenté à

M. Dominique Grenier, M. Luc Lamontagne et M. Abdelhakim Bendada

<i>matricule</i>	<i>nom</i>
910 058 073	Émile Arsenault
908 190 985	Philippe Bourdages
910 098 468	Pierre-Luc Buhler
998 107 355	Diane Fournier
908 318 388	Olivier Sylvain
910 055 897	Daniel Thibodeau
910 097 879	Francis Valois

Université Laval  
4 avril 2013

# Table des matières

<b>Table des figures</b>	<b>ii</b>
<b>Liste des tableaux</b>	<b>iii</b>
<b>1 Diagrammes de classes</b>	<b>1</b>
<b>2 Diagrammes de séquences</b>	<b>3</b>
<b>3 Plan d'intégration</b>	<b>7</b>
<b>4 Registre de risques</b>	<b>9</b>
<b>5 Vision 1<sup>ère</sup> itération</b>	<b>13</b>
5.1 Localisation avec la caméra embarquée . . . . .	13
5.1.1 Localisation . . . . .	13
5.1.1.1 Transformations . . . . .	13
5.1.1.2 Localisation des points . . . . .	14
5.1.2 Orientation et angle par rapport au mur . . . . .	14
5.1.3 Extraire les contours d'un sudocube . . . . .	14
5.2 Localisation avec la Kinect . . . . .	15
5.2.1 Transformation des distances . . . . .	15
5.2.2 Localisation de la Kinect à l'aide de la calibration . . . . .	16
5.2.3 Détection des obstacles . . . . .	17
5.2.4 Détection du robot . . . . .	18
<b>6 Schémas électroniques 1<sup>ère</sup> itération</b>	<b>19</b>

# Table des figures

1.1	Figure présentant le diagramme de classes du projet . . . . .	2
2.1	Diagramme des séquences présentant les liens entre l'utilisateur et la kinocto	3
2.2	Diagramme des séquences présentant les liens entre la kinocto et son environnement afin de déterminer sa position ainsi que le décodage de l'antenne avec les mouvements associés pour arriver . . . . .	4
2.3	Diagramme des séquences présentant les liens entre la kinocto et la station de base pour la transmission et l'affichage ainsi que la lecture et le décodage du sudocube avec les mouvements nécessaires pour y arriver . . . . .	5
2.4	Diagramme des séquences présentant les liens entre la kinocto et la table de jeu dans l'optique du déplacement de celui-ci pour effectuer le dessin du chiffre contenu dans la case rouge du sudocube . . . . .	6
3.1	Figure présentant le plan d'intégration . . . . .	8
5.1	Schéma représentant la table et les différentes mesures obtenues avec la Kinect pour un point quelconque . . . . .	16
5.2	Schéma représentant la table et les vecteurs nécessaires à la mesure de l'angle de la Kinect . . . . .	17
5.3	Schéma représentant la table et les vecteurs nécessaires à la localisation de la Kinect . . . . .	18
6.1	Figure présentant les plans de l'alimentation 5V pour les périphériques . . .	19
6.2	Figure présentant une photo de l'alimentation 24V pour les périphériques . .	19
6.3	Figure présentant les plans du récepteur Manchester . . . . .	20
6.4	Figure présentant les plans du circuit de commande du préhenseur . . . . .	20
6.5	Figure présentant les plans du circuit de commande du préhenseur . . . . .	20

# Liste des tableaux

4.1	Registre de risques partie 1 . . . . .	10
4.2	Registre de risques partie 2 . . . . .	11
4.3	Registre de risques partie 3 . . . . .	12

# Chapitre 1

## Diagrammes de classes

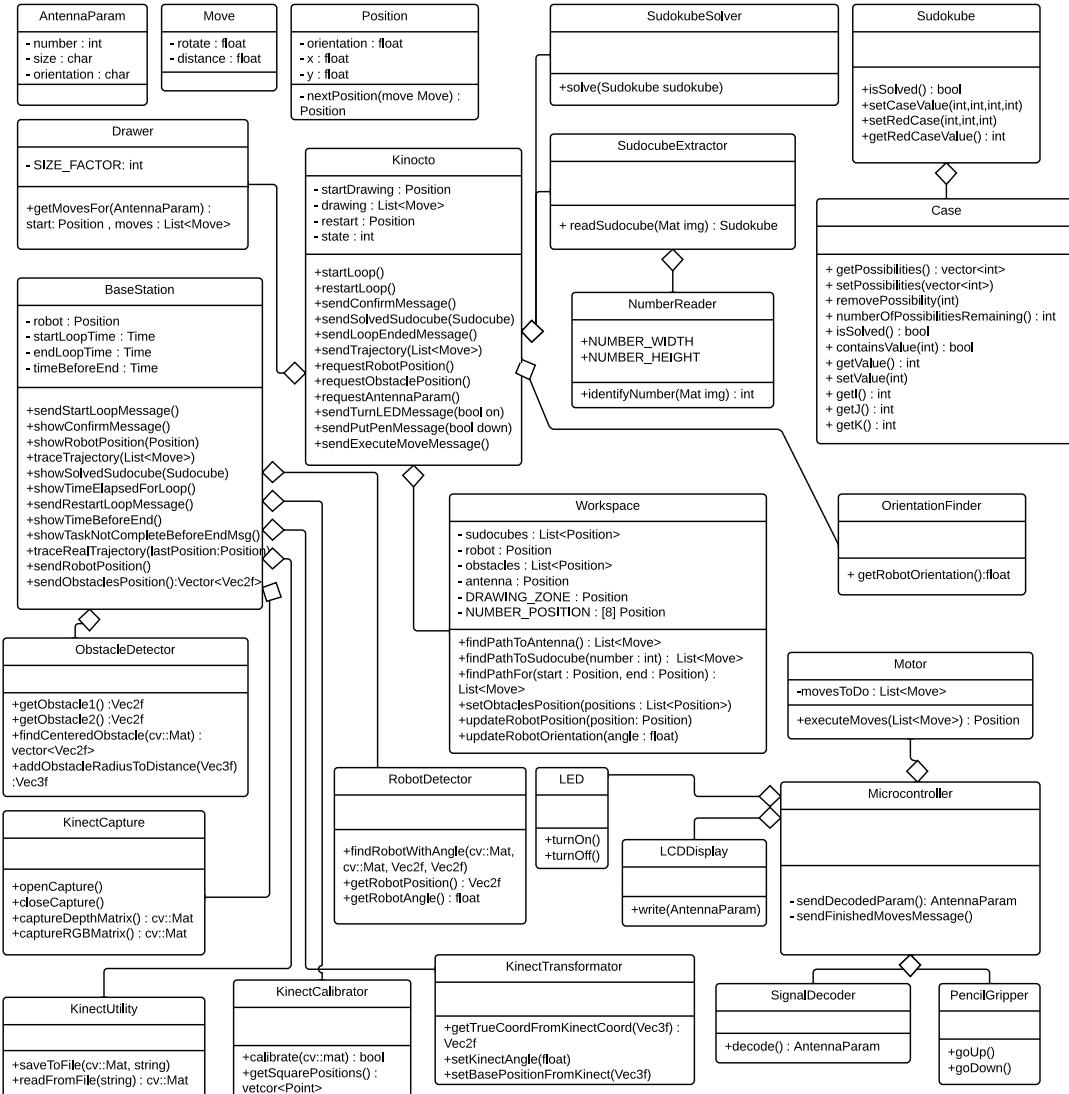


Figure 1.1 – Figure présentant le diagramme de classes du projet

# Chapitre 2

## Diagrammes de séquences

Ce chapitre présente les différentes figures associées au diagramme des séquences. Ce diagramme est séparé selon cinq portions relatives aux fonctions particulières du robot. La figure 2.1 présente les liens entre l'utilisateur et la kinocto. La figure 2.2 présente les liens entre la kinocto et son environnement afin de déterminer sa position ainsi que le décodage de l'antenne avec les mouvements associés pour arriver. La figure 2.3 présente les liens entre la kinocto et la station de base pour la transmission et l'affichage ainsi que la lecture et le décodage du sudocube avec les mouvements nécessaires pour y arriver. La figure 2.4 présente les liens entre la kinocto et la table de jeu dans l'optique du déplacement de celui-ci pour effectuer le dessin du chiffre contenu dans la case rouge du sudocube.

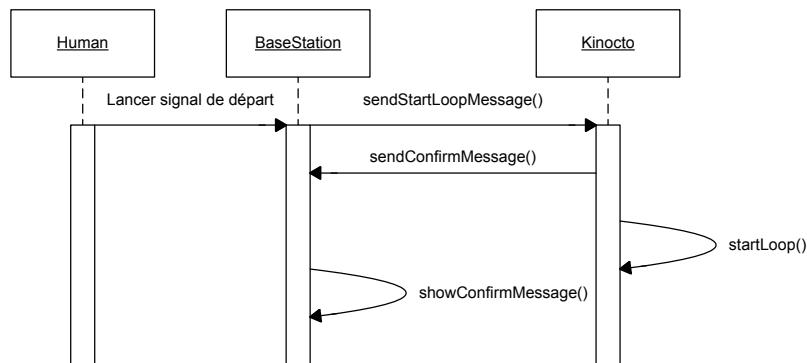


Figure 2.1 – Diagramme des séquences présentant les liens entre l'utilisateur et la kinocto

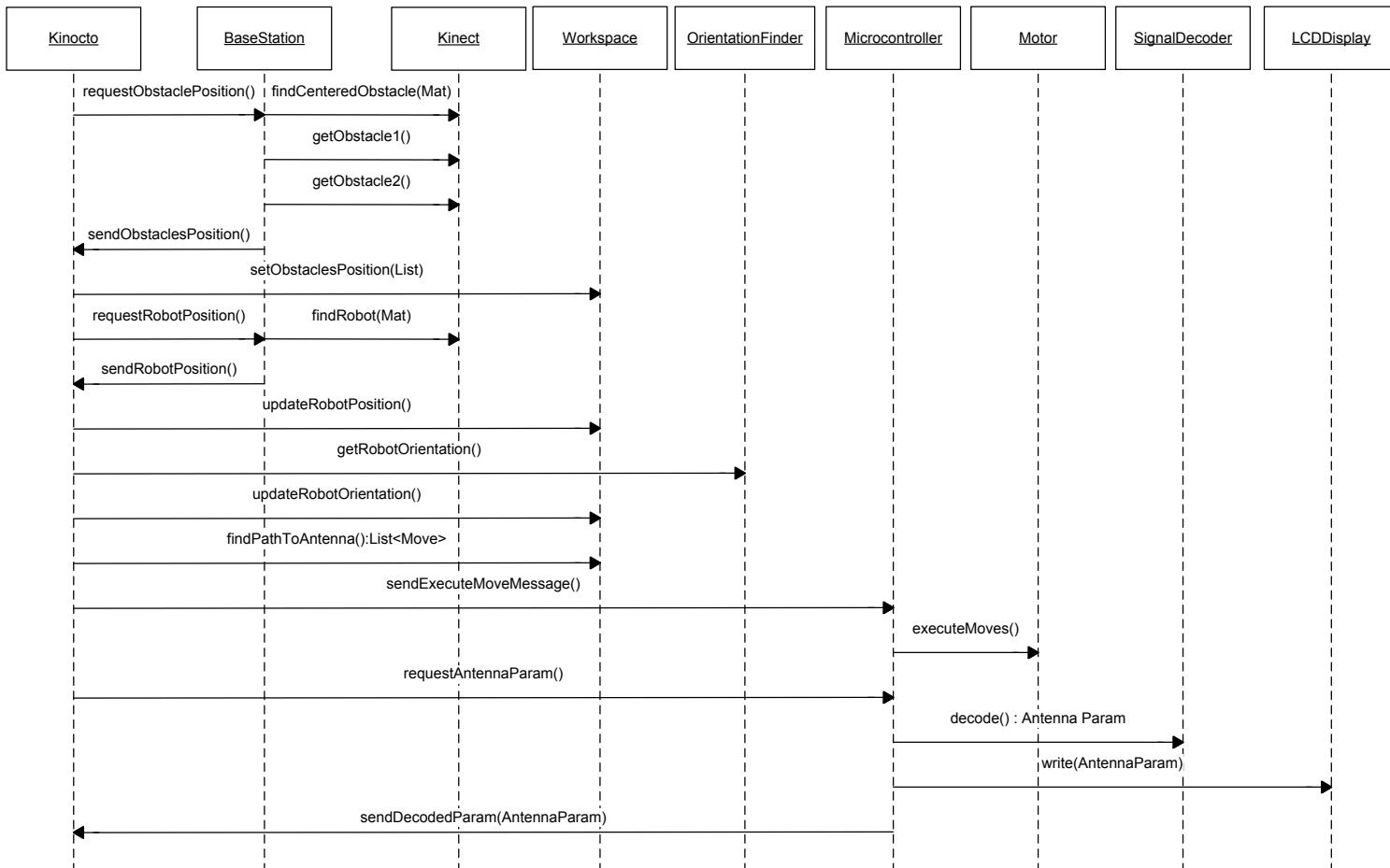


Figure 2.2 – Diagramme des séquences présentant les liens entre la kinecto et son environnement afin de déterminer sa position ainsi que le décodage de l'antenne avec les mouvements associés pour arriver



Figure 2.3 – Diagramme des séquences présentant les liens entre la kinecto et la station de base pour la transmission et l'affichage ainsi que la lecture et le décodage du sudocube avec les mouvements nécessaires pour y arriver

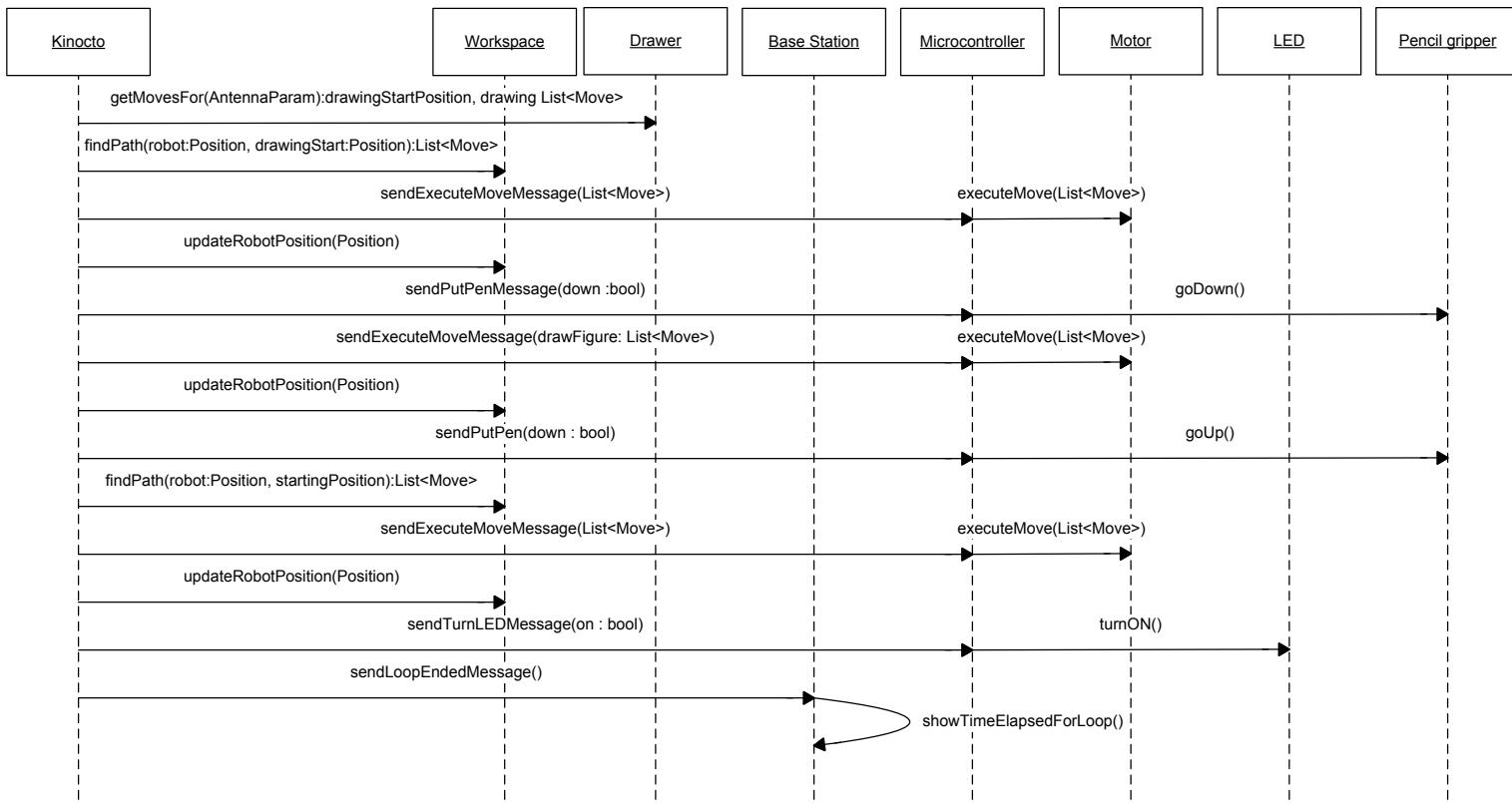


Figure 2.4 – Diagramme des séquences présentant les liens entre la kinecto et la table de jeu dans l'optique du déplacement de celui-ci pour effectuer le dessin du chiffre contenu dans la case rouge du sudocube

# Chapitre 3

## Plan d'intégration

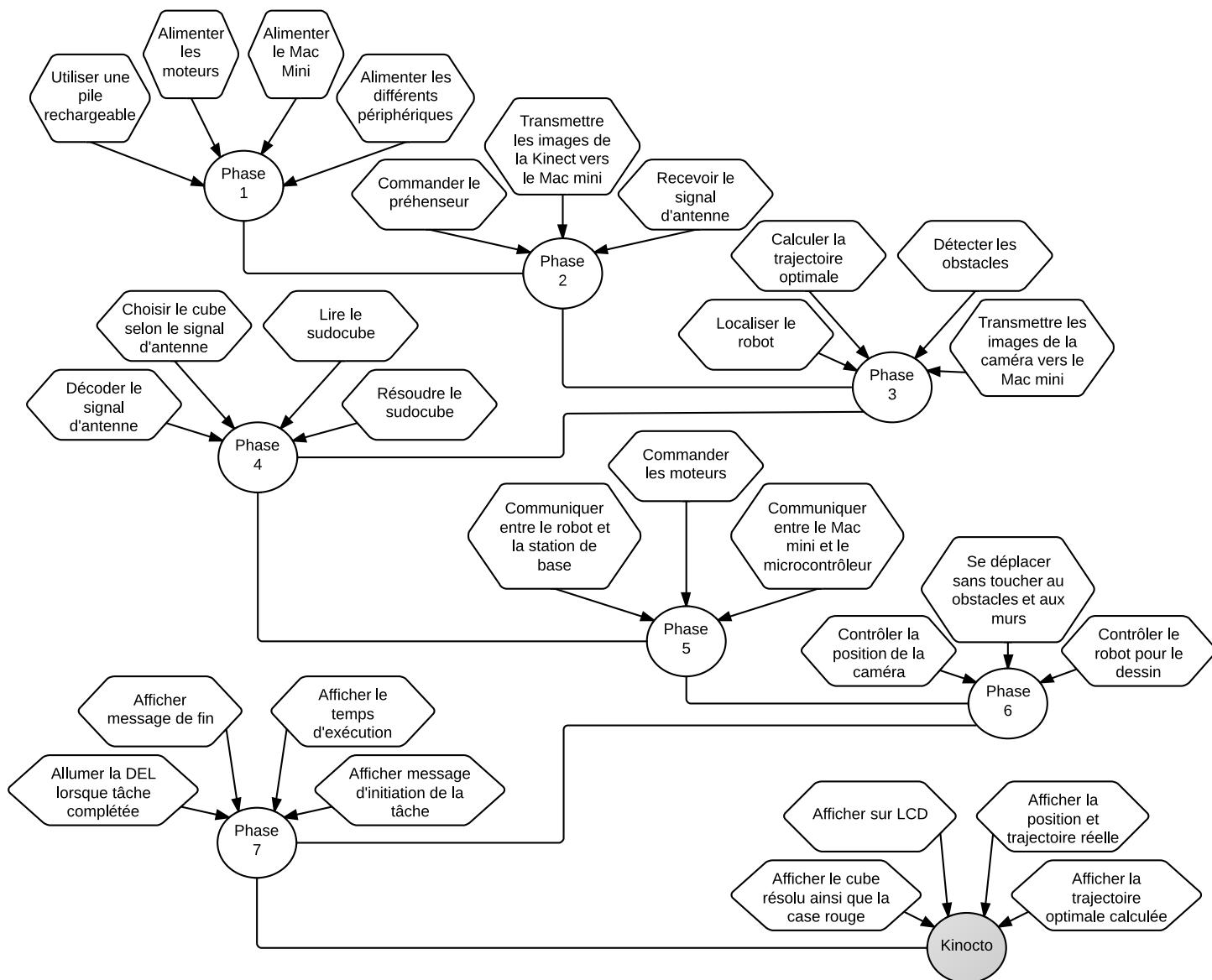


Figure 3.1 – Figure présentant le plan d'intégration

# Chapitre 4

## Registre de risques

Tableau 4.1 – Registre de risques partie 1

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
1	Bris de la batterie suite à une usure prématuée	5	Plus d'autonomie énergétique pour le robot, opération impossible	L'ensemble du système sera inopérable	5	60	Formation des utilisateurs à l'endroit de l'utilisation. Apposition d'un capteur de tension. Dispositifs de protection (fusibles). Utiliser un système de recharge adéquat pour la batterie.	É. Arsenault
2	Utilisation entraînant l'alimentation hors de sa zone de fonctionnement	5	Bris d'un ou des système d'alimentation. Les systèmes auxiliaires, le Mac mini ou les moteurs ne seront pas alimentés. Le système ne sera pas fonctionnel	Une partie ou l'ensemble du système sera inopérable	5	25	Utilisateur de connecteurs protégés (d'un seul sens possible). Dispositifs de protection (fusibles), surdimensionnement des composantes d'alimentation. Achat de pièces de rechange.	D. Thibodeau
3	Bris du crayon lors du dessin	4	Le dessin ne pourra être complété	La portion dessin sera partiellement complète	20	2	Tests rigoureux et optimisation du choix de crayon avant la compétition	É. Arsenault
4	Bris du système de préhension	5	Le dessin ne pourra être complété et selon le moment du bris, la trajectoire du robot peut être affectée	La portion dessin sera partiellement complète et la trajectoire sera déviée	5	10	Tests rigoureux et essais multiples pour vérifier la stabilité en température lors du fonctionnement	É. Arsenault
5	Mauvais branchement avec le microcontrôleur	4	Bris d'une portion ou de la totalité du microcontrôleur. Le bris d'une portion du microcontrôleur empêche l'exécution de la tâche dans son ensemble et peut occasionner des bris dans les systèmes reliés	Un robot qui ne peut pas se déplacer correctement, qui n'allume pas la DEL ou qui n'active pas le préhenseur	5	100	Isolation des entrées et sorties avec des dispositifs de protection (diode), limiteur de courant	D. Fournier

Tableau 4.2 – Registre de risques partie 2

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
6	Mauvaise utilisation des entrées/sorties ou des ports d'alimentation du Mac mini	5	Bris d'une portion ou de la totalité du Mac mini. Une portion ou l'ensemble des auxiliaires ne fonctionnera pas correctement. Le système sera partiellement fonctionnel	Robot incapable d'effectuer l'ensemble de la tâche	5	600	Apposition de protections électriques (fusibles et interrupteurs) sur l'étage d'alimentation du Mac mini. Fixation robuste du Mac mini sur le robot. Protection du port USB du Mac mini en n'utilisant pas le fil d'alimentation.	F. Valois
7	Collision de la caméra web avec les obstacles lors des déplacements ou mauvaise manipulation de celle-ci.	4	Bris de la caméra web. Le bris de la caméra empêche la vision des cubes	Si la caméra ne peut voir le sude cube, on ne peut trouver le chiffre dans la case rouge et effectuer le bon dessin	5	80	Storage adéquat de la caméra, protection d'alimentation (fusible), limiter les chocs contre les obstacles	D. Thibodeau
8	Problème de communication entre le Mac mini et la station de base	5	Les informations requises ne pourront être transmises correctement, on perd l'information sur le comportement du robot ainsi que sa position. Le robot ne pourra pas se localiser initialement et en temps réel.	Le robot ne remplira pas les exigences d'affichage sur la station de base, le robot ne recevra aucune position de la Kinect	5		Tests répétés pour la transmission et de la réception de l'information en temps réel entre le Mac mini et la station de base	P. Bourdages
9	Bris du système d'exploitation du Mac mini	4	La portion logicielle du robot et le traitement seront absents. Le système ne sera pas fonctionnel	Robot incapable d'accomplir un traitement de tâches	30		Clonage d'une version fonctionnelle et stable du système d'exploitation	P. Buhler

Tableau 4.3 – Registre de risques partie 3

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
10	Problème de communication entre le Mac mini et le microcontrôleur	5	L'ensemble des auxiliaires ne fonctionnera pas correctement. Le système sera non fonctionnel	Robot incapable de se déplacer selon la trajectoire prévue et d'effectuer la tâche	5		Tests répétés pour la transmission et de la réception de l'information entre le Mac mini et le microcontrôleur	D. Fournier
11	Contact avec un ou des obstacles	3	Bris du système et déviation de trajectoire possible	Un robot qui entre en contact avec les obstacles ne remplit pas les exigences du projet	20		Tests rigoureux sur les déplacements et marge de sécurité importante pour le contournement.	O. Sylvain
12	Problème d'identification du robot et de l'environnement (vision) par la Kinect	5	Correction de la trajectoire erronée, risque de rencontrer les obstacles, mauvaise trajectoire calculée	La trajectoire réelle ne sera pas optimale et le robot risque de rencontrer des obstacles	10		Tests rigoureux et utilisation de plusieurs images pour le calcul.	F. Valois
13	Perturbations de l'environnement du robot (irrégularités sur la table ou dans l'éclairage)	2	La trajectoire du robot pourrait être déviée si la vision est entachée par une mauvaise luminosité ou une irrégularité dans la table	La trajectoire parcourue par le robot ne sera pas idéale	10		Tests rigoureux des algorithmes de vision et d'asservissement, essais avec ajout d'irrégularités	D. Thibodeau
14	Départ de l'un des membres de l'équipe	3	La quantité de tâches des membres restants de l'équipe devra être augmentée.	Des détails et des ajustements de pointes nécessitant plus de temps ne seront pas réalisés	5		S'assurer d'une bonne communication dans l'équipe et d'un bon transfert de connaissances	D. Thibodeau

# Chapitre 5

## Vision 1<sup>ère</sup> itération

### 5.1 Localisation avec la caméra embarquée

La localisation avec la caméra embarquée est un système de localisation d'appoint dans le projet Kinocto. La classe Localisation doit recevoir une image et une orientation grossière selon le nord, sud, est ou ouest par rapport à la table. Elle doit ensuite être initialisée en fournissant un fichier .xml contenant les paramètres de calibration. Ces paramètres comprennent les paramètres intrinsèques de la caméra obtenus lors de la calibration, les paramètres liés à la distortion de l'image par la lentille, les paramètres extrinsèques qui permettent de lier les points des images prises par la caméra à leur position par rapport à un système de référence virtuel ainsi que les paramètres qui permettent de recadrer ces coordonnées par rapport au centre du robot.

Avec l'image et les paramètres provenant de la calibration, les méthodes de localisation, de détermination de l'orientation et de mesure de l'angle par rapport au mur peuvent être utilisées.

#### 5.1.1 Localisation

##### 5.1.1.1 Transformations

Pour effectuer la localisation du robot, il faut trouver dans l'image reçue au moins deux points identifiables dont les coordonnées par rapport à la table sont connues. Dans le projet Kinocto, ces points incluent les coins de la tables, les coins inférieurs des blocs de couleur, les coins du carré vert dessiné sur la table. Pour passer du système de coordonnée du robot à celui de la table, il faut résoudre le système d'équation suivant, où les points  $P_{1A}$  et  $P_{2A}$  sont les points dans le systèmes de la table,  $P_{1R}$  et  $P_{2R}$  sont les mêmes points dans le système du robot et  $t_X$  et  $t_Y$  sont les coordonnées du robot dans le système de la table :

$$X_{1A} = X_{1R}\cos(\theta) - Y_{1R}\sin(\theta) + t_X \quad (5.1)$$

$$Y_{1A} = X_{1R}\sin(\theta) + Y_{1R}\cos(\theta) + t_Y \quad (5.2)$$

$$X_{2A} = X_{2R}\cos\theta - Y_{2R}\sin(\theta) + t_X \quad (5.3)$$

$$Y_{2A} = X_{2R}\sin(\theta) + Y_{2R}\cos(\theta) + t_Y \quad (5.4)$$

En soustrayant l'équation 5.3 de l'équation 5.1 ainsi que l'équation 5.4 de l'équation 5.2, on élimine  $t_X$  et  $t_Y$  et on se retrouve avec deux équations que l'on peut combiner pour obtenir une équation à une inconnue,  $\theta$ . Il s'agit d'une équation de la forme  $a\cos(\theta) + b\sin(\theta) = c$ . Ce type d'équation peut être résolu en considérant la relation suivante :

$$a\cos(\theta) + b\sin(\theta) = R\cos(\theta - \alpha) \quad (5.5)$$

Où  $R = \sqrt{a^2 + b^2}$  et  $\tan(\alpha) = \frac{b}{a}$ . En isolant  $\theta$ , on obtient donc :

$$\theta = \cos^{-1}\left(\frac{c}{\sqrt{a^2 + b^2}}\right) + \tan^{-1}\left(\frac{b}{a}\right) \quad (5.6)$$

Il suffit ensuite d'utiliser  $\theta$  dans deux des équations initiales pour retrouver les coordonnées du robot selon le système de référence de la table  $t_X$  et  $t_Y$ .

### 5.1.1.2 Localisation des points

#### 1. Coins de table

Les points correspondant aux coins de table sont trouvés dans l'image en effectuant d'abord une segmentation sur le noir et en utilisant ensuite l'algorithme des lignes de Hough pour retrouver les lignes de bas de mur. L'intersection entre les lignes de bas de mur constitue le coin de la table. Un intérêt de cette méthode est qu'elle permet de retrouver les coins de table même lorsqu'un obstacle se trouve devant celui-ci.

#### 2. Coins du bas des blocs de couleur

Les coins du bas des blocs de couleurs peuvent être retrouvés en trouvant les intersections entre les lignes de bas de mur et la ligne du bas du bloc de couleur, trouvée après segmentation sur le bleu et l'orange.

#### 3. Coins du carré vert

Les coins internes du carré vert peuvent être retrouvés en segmentant sur le vert et en retrouvant les intersections entre les lignes avec la méthode décrite plus haut.

## 5.1.2 Orientation et angle par rapport au mur

L'angle par rapport au mur peut être utile pour réorienter le robot pour la lecture des sudokus. Il est facilement obtenu en utilisant la ligne de bas de mur détectée précédemment. L'orientation peut être déduite en utilisant cet angle et l'orientation fournie à l'initialisation.

## 5.1.3 Extraire les contours d'un sudocube

La première étape d'extraction consiste à segmenter par couleur verte afin d'isoler le cadre du sudocube. Puis, un algorithme de détection des contours est appliqué pour trouver deux polygones rectangulaires. Ces polygones sont les bordures intérieures et extérieures du cadre. L'aire des rectangles est calculée pour vérifier que les polygones sont assez grands pour être ceux du cadre.

Ensuite, à partir du polygone intérieur du cadre on isole une sous région de l'image principale. L'image obtenue est convertie en noir et blanc afin d'appliquer à nouveau un algorithme de contours afin de trouver les cases du sudocube. On sélectionne les polygones résultants selon leur aire. Si jamais le nombre de cases sélectionné n'est pas de 47 on recommence avec un seuil de tolérance différent pour l'algorithme de contour. Le polygone de la case rouge est obtenue en segmentant par couleur rouge et appliquant à nouveau un algorithme de contour.

Finalement, les images des cases sont triées en fonction de leur position en X et en Y dans l'image du sudocube et un algorithme de reconnaissance des caractères appliqué sur toutes les cases afin d'identifier les numéros. L'algorithme de reconnaissance utilise la technique KNearest. Les chiffres trouvés sont ajoutés dans la structure de données du sudocube et la position de la case rouge est spécifiée.

## 5.2 Localisation avec la Kinect

Pour aider au déplacement du robot, la Kinect a été utilisée afin de détecter la position des obstacles, la position du robot lui-même ainsi que sa position angulaire par rapport au point (0,0) de notre représentation cartésienne de la table. Toutefois, comme la Kinect est située à l'extérieur de la table de jeu, il faut une translation et une rotation des données obtenues afin de positionner les objets avec exactitude. Les 3 sections qui suivent expliquent en détail les différentes parties de l'algorithme de vision de la Kinect.

### 5.2.1 Transformation des distances

En premier lieu, il est important de clarifier quelles distances il est possible d'obtenir à l'aide de la Kinect. Le «Framework» OpenNI est capable de retourner 2 types de distances pour chacun des points dans l'image infrarouge obtenue de la Kinect. Le premier type de distances retourné est la longueur, en mètres, entre l'objectif et un point quelconque sur l'image. Le second type est dérivé du premier et correspond aux composantes X, Y et Z du vecteur de longueur entre l'objectif et la cible. Pour aider à la compréhension, les deux types de distances peuvent être représentés par les lignes vertes sur l'image 5.1. Toutefois, comme l'origine de notre représentation cartésienne de la table est située sur le coin inférieur droit de la table et que la Kinect n'est pas située à cet endroit, il est nécessaire d'obtenir la distance X et Y (lignes bleues sur l'image 5.1) entre l'origine et le point ciblé sur l'image obtenue de la Kinect. Pour y arriver, il suffit d'obtenir la position X et Y (lignes rouges sur l'image 5.1) de la lentille infrarouge de la Kinect ainsi que l'angle de la Kinect avec l'axe X de la table. Pour obtenir ces valeurs, nous utilisons une calibration simple présentée à la section 5.2.2. Une fois la calibration effectuée, il est possible d'effectuer la transformation des distances de la Kinect vers les composantes X et Y recherchées à l'aide de règles de trigonométrie simples (Translation et Rotation). Toutefois, la Kinect possède une erreur de mesure qui augmente avec la distance. C'est pourquoi l'algorithme va toujours posséder une incertitude entre 1 et 3 cm pour chacun des points mesurés.

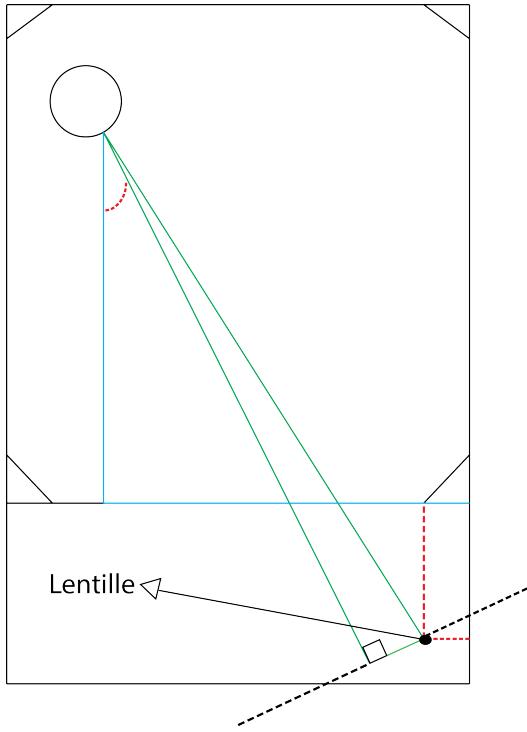


Figure 5.1 – Schéma représentant la table et les différentes mesures obtenues avec la Kinect pour un point quelconque

### 5.2.2 Localisation de la Kinect à l'aide de la calibration

Le but de la calibration est d'obtenir l'angle de la Kinect avec l'axe X de la table ainsi que la position de son objectif en XY par rapport au coin inférieur droit de la zone de jeu qui constitue notre point (0,0). Pour y arriver, un système, qui possède des dimensions connues, va positionner une plaque constituée d'un "Chessboard" sur la table à un endroit pré-déterminé (ligne noire sur la table dans l'image 5.2). Par la suite, une image RGB de la table est capturée à l'aide de la Kinect et les 2 points les plus éloignés horizontalement sur le "Chessboard" sont localisés. Ces 2 points sont reportés sur l'image de distances correspondante pour obtenir la distance XYZ de chacuns des 2 points (lignes vertes sur l'image 5.2). À l'aide de ces deux distances, les dimensions du petit triangle entre les deux points de la plaque de calibration sont trouvées (le triangle se situe en dessous de la plaque de calibration sur l'image 5.2). En ayant les dimensions du triangle, il est possible de trouver l'angle de la Kinect car celui-ci est le même que l'angle interne droit du triangle. Il suffit d'utiliser l'équation suivante avec les dimensions obtenues précédemment :

$$\text{Angle} = \arctan \left( \frac{\text{oppose}}{\text{adjacent}} \right) \quad (5.7)$$

Pour obtenir la position de la Kinect, il est nécessaire de connaître la position réelle, par rapport au point (0,0), des 2 points utilisés pour la mesure de l'angle. En effectuant la

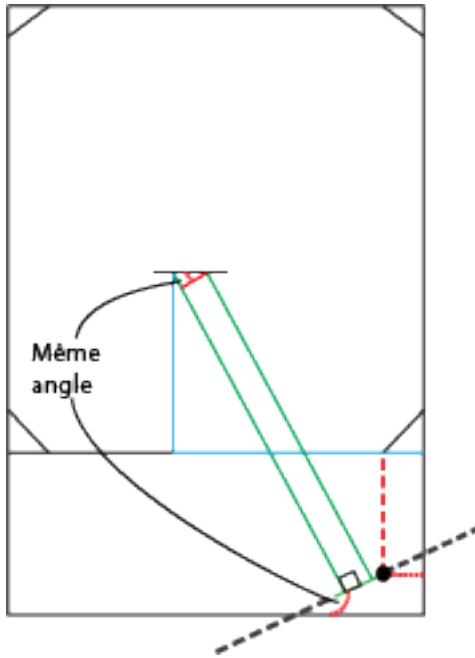
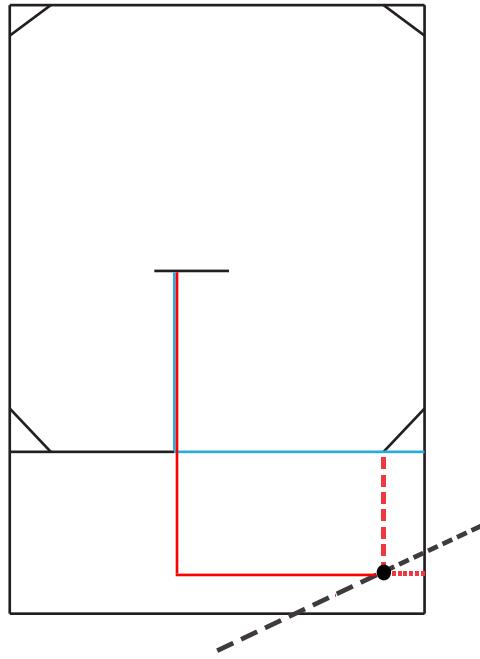


Figure 5.2 – Schéma représentant la table et les vecteurs nécessaires à la mesure de l'angle de la Kinect

rotation des distances de chacun des points données par la Kinect pour ramener les mesures dans le plan XY de la table, les distances XY entre les points et l'objectif sont obtenues (lignes rouges pleines sur l'image 5.3). En soustrayant les distances précédemment obtenues aux distances réelles mesurées (lignes bleues sur l'image 5.3), la position de la Kinect est obtenues (lignes rouges pointillées sur l'image 5.3).

### 5.2.3 Détection des obstacles

Comme il est possible d'obtenir n'importe quelle distance entre l'origine et un point quelconque sur l'image, un algorithme de recherche d'obstacles a été créé. Sachant que les obstacles sont situés dans une zone précise sur la table et que ceux-ci font 40cm de hauteur, il suffit de rechercher tout objet de cette hauteur situé dans la zone pré définie. La Kinect retourne la hauteur de chacun des points sur l'image infrarouge et permet de localiser les obstacles. De plus, à l'aide de calculs statistiques, l'algorithme est en mesure de trouver les obstacles lorsqu'ils sont presque alignés ou obstrués par un autre objet comme le robot. Comme cet algorithme dépend entièrement de l'algorithme de transformation des distances, les positions obtenues pour chacun des obstacles possèdent la même incertitude de 1 à 3 cm sur chaque mesure de distance effectuée. Pour améliorer la validité des mesures obtenues, une moyenne sur 3 images de distance est effectuée. En ce qui concerne l'efficacité de l'algorithme, différents tests montrent un temps de calcul d'environ 30 ms sur un Core 2 Duo 2.4 GHz pour obtenir la position du centre des deux obstacles.



*Figure 5.3 – Schéma représentant la table et les vecteurs nécessaires à la localisation de la Kinect*

#### 5.2.4 Détection du robot

En ce qui concerne la détection du robot, un algorithme semblable à la détection des obstacles a été utilisé. Comme le robot possède une hauteur d'environ 20cm et une profondeur semblable, il est possible d'isoler, dans la matrice de distance, un carré qui correspond aux dimensions du robot. En plus de cette méthode, 2 "Chessboard" de couleurs différentes sont placés sur 2 faces opposées du robot. En recherchant les carrés sur le robot, il est possible de déterminer sa position comme l'algorithme précédent mais en plus d'obtenir son angle car la distance entre les points sont connus ainsi que son orientation car les "Chessboard" ne sont pas de la même couleur. Pour contrer l'erreur de position causée par la Kinect, la méthode de moyenne des distances utilisée pour la détection des obstacles est aussi appliquée à la détection du robot. Le temps de calcul est un peu plus élevé que la détection du robot et se situe aux alentours de 50 à 60 ms.

# Chapitre 6

## Schémas électriques 1<sup>ère</sup> itération

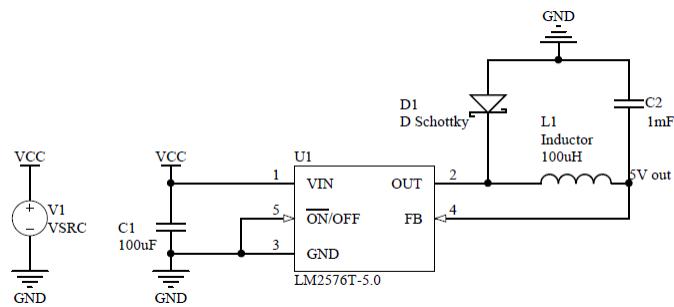


Figure 6.1 – Figure présentant les plans de l'alimentation 5V pour les périphériques

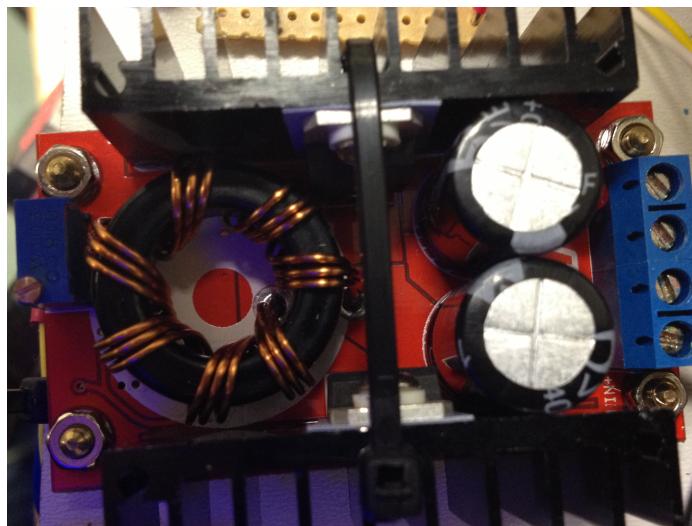


Figure 6.2 – Figure présentant une photo de l'alimentation 24V pour les périphériques

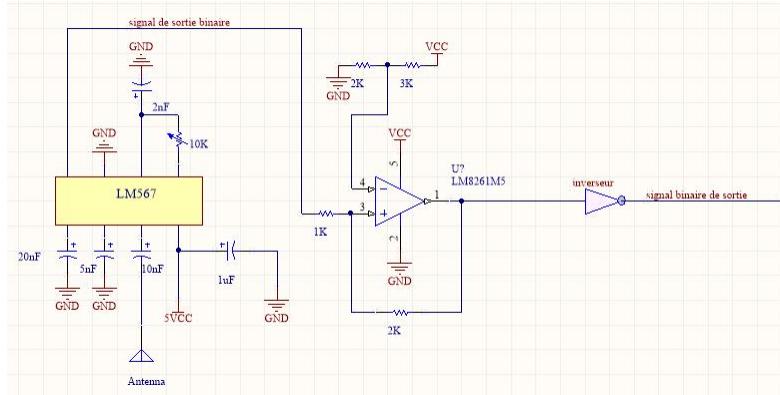


Figure 6.3 – Figure présentant les plans du récepteur Manchester

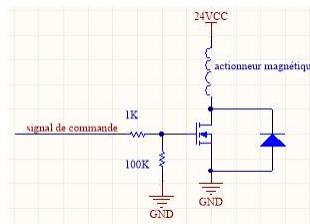


Figure 6.4 – Figure présentant les plans du circuit de commande du pré-henseur

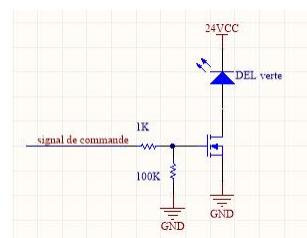


Figure 6.5 – Figure présentant les plans du circuit de commande du pré-henseur