RobotLib

INF1995 - Projet

6/12/2013

Chapter 1

Module Index

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Module Documentation

Module Documentation

2.1 Aleatoire

Permet de gerer les deplacements aleatoires.

Functions

- void initTimerAleatoire ()

  Permet d'initialiser le timer2 pour creer un pseudo aleatoire.
- Direction prendreValeurAleatoire ()

  Retourne une valeur aleatoire de type Direction. (50% des changes que le resultat soit direction_continue et 25% des chances pour chancun que le resultat soit direction_left ou direction_right).

2.1.1 Detailed Description

Permet de gerer les deplacements aleatoires.

2.1.2 Function Documentation

2.1.2.1 void initTimerAleatoire ( )

Permet d'initialiser le timer2 pour creer un pseudo aleatoire.

2.1.2.2 Direction prendreValeurAleatoire ( )

Retourne une valeur aleatoire de type Direction. (50% des changes que le resultat soit direction_continue et 25% des chances pour chancun que le resultat soit direction_left ou direction_right).

2.2 Can

Permet de gerer la conversion de analogique a numerique.

Classes

- class can

  Le constructeur initialise le convertisseur. Une lecture enclanche une conversion et le resultat est retourne sur 16 bits.

2.2.1 Detailed Description

Permet de gerer la conversion de analogique a numerique.

2.3 Del

Permet de controller la del.

Functions

- void clignoterDel (volatile uint8_t ∗ddr, volatile uint8_t ∗port, const uint8_t &pin, const uint16_t &nfois, const uint8_t &color)

  Permet de faire clignoter un del, soit en rouge ou en vert.
- void allumerDel (volatile uint8_t ∗ddr, volatile uint8_t ∗port, const uint8_t &pin, const uint16_t &temps_ms, const uint8_t &color)

  Permet d'allumer un del pendant un certain temps, soit en rouge ou en vert.
- void garderAllumerDel (volatile uint8_t ∗ddr, volatile uint8_t ∗port, const uint8_t &pin, const uint8_t &color)

  Permet d'allumer un del, soit en rouge ou en vert.
- void eteindreDel (volatile uint8_t ∗ddr, volatile uint8_t ∗port, const uint8_t &pin)

  Permet d'eteindre un del.

2.3.1 Detailed Description

Permet de controller la del.

2.3.2 Function Documentation

2.3.2.1 void allumerDel (volatile uint8_t ∗ *ddr,* volatile uint8_t ∗ *port,* const uint8_t & *pin,* const uint16_t & *temps_ms,* const uint8_t & *color*)

Permet d'allumer un del pendant un certain temps, soit en rouge ou en vert.

Parameters

ddr	Permet de choisir le port de configuration des entrees sorties.
port	Choisir le port sur lequel le del se trouve.
pin	Choisir la premiere pin ou le del est connecte.
temps	Choisir le nombre de temps que le del sera allume (par tranche de 100 millisecondes).
color	Choisir si le del clignote en rouge ou en vert. (DEL_VERT, DEL_ROUGE)

2.3.2.2 void clignoterDel (volatile uint8_t ∗ *ddr,* volatile uint8_t ∗ *port,* const uint8_t & *pin,* const uint16_t & *nfois,* const uint8_t & *color*)

Permet de faire clignoter un del, soit en rouge ou en vert.

Parameters

ddr	Permet de choisir le port de configuration des entrees sorties.
port	Choisir le port sur lequel le del se trouve.
pin	Choisir la premiere pin ou le del est connecte (0-7).
nfois	Choisir le nombre de fois que le del clignotera.
color	Choisir si le del clignote en rouge ou en vert. (DEL_VERT, DEL_ROUGE)

2.3.2.3 void eteindreDel (volatile uint8_t ∗ *ddr,* volatile uint8_t ∗ *port,* const uint8_t & *pin*)

Permet d'eteindre un del.

Parameters

ddr	Permet de choisir le port de configuration des entrees sorties.
port	Choisir le port sur lequel le del se trouve.
pin	Choisir la premiere pin ou le del est connecte.

2.3.2.4 void garderAllumerDel (volatile uint8_t * *ddr*, volatile uint8_t * *port*, const uint8_t & *pin*, const uint8_t & *color*)

Permet d'allumer un del, soit en rouge ou en vert.

Parameters

ddr	Permet de choisir le port de configuration des entrees sorties.
port	Choisir le port sur lequel le del se trouve.
pin	Choisir la premiere pin ou le del est connecte.
color	Choisir si le del clignote en rouge ou en vert. (DEL_VERT, DEL_ROUGE)

2.4 Lcd

Permet de gerer l'ecriture sur l'ecran lcd pour debugger.

Functions

- void writeValue (LCM *lcd, const int &value, const uint8_t &value_number)
  WriteValue (INT)
- void writeValue (LCM *lcd, const float &value, const uint8_t &value_number)
  WriteValue (FLOAT).
- void writeValue (LCM *lcd, const char *txt, const uint8_t &value_number)
  WriteValue (STRING).
- uint16_t writeVx (LCM *lcd, char *tmp, const uint8_t &value_number)
  writeVx est une fonction interne utilisee par writeValue(..) pour afficher Vx de 0 a 3.

2.4.1 Detailed Description

Permet de gerer l'ecriture sur l'ecran lcd pour debugger.

2.4.2 Function Documentation

2.4.2.1 void writeValue (LCM * *lcd*, const int & *value*, const uint8_t & *value_number*)

WriteValue (INT)

Parameters

lcd	Classe LCM pour utiliser la fonction write sur le bon port.
value	Valeur a etre ecrite.
value_number	Numero de la valeur soit (0, 1, 2, 3) pour V0 a V3.

2.4.2.2 void writeValue (LCM * *lcd*, const float & *value*, const uint8_t & *value_number*)

WriteValue (FLOAT).

Parameters

lcd	Classe LCM pour utiliser la fonction write sur le bon port.
value	Valeur a etre ecrite entre [0.00 et 999.99].
value_number	Numero de la valeur soit (0, 1, 2, 3) pour V0 a V3.

2.4.2.3 void writeValue (LCM * *lcd*, const char * *txt*, const uint8_t & *value_number*)

WriteValue (STRING).

Parameters

lcd	Classe LCM pour utiliser la fonction write sur le bon port.
value	String de 5 characteres maximum.
value_number	Numero de la valeur soit (0, 1, 2, 3) pour V0 a V3.

2.4.2.4 uint16_t writeVx (LCM * *lcd*, char * *tmp*, const uint8_t & *value_number*)

writeVx est une fonction interne utilisee par writeValue(..) pour afficher Vx de 0 a 3.

Parameters

lcd	Classe LCM pour utiliser la fonction write sur le bon port.
tmp	Buffer pour utiliser itoa dans la fonction. (la valeur du buffer sera alteree).
value_number	Numero de la valeur soit (0, 1, 2, 3) pour V0 a V3.

Todo Mettre tous dans un seul tableau avant d'appeler write.

2.5 LineFollower

Permet de suivre le ligne, de detecter les intersections et de tourner.

Classes

- class LineFollower

  Permet la gestion des lignes et intersections. Lorsque le capteur tttraverse une intersection, une fonction passee en parametre par pointeur est appelee. Cette fonction, definie par l'utilisateur, permet de choisir la direction qui le robot prendra. Si les fonctions pour tourner ne sont pas appelees, le robot va continuer a avancer. La classe permet egalement de tourner automatiquement au intersection aux coins.

2.5.1 Detailed Description

Permet de suivre le ligne, de detecter les intersections et de tourner.

2.6 Audio

Permet de faire jouer un son ou une note midi.

Functions

- void jouerSon (const uint8_t &v)

  Permet de faire jouer un son.
- void jouerNote (const uint8_t &midi_note)

  jouerNote permet de jouer une note midi de 45 a 81.
- void silence ()

  silence permet d'arreter le piezo de jouer la note en cour.
- void timer0_init (void)

  timer0_init permet d'initialiser les bonnes valeurs de prescale et d'initiation du Timer0.

2.6.1 Detailed Description

Permet de faire jouer un son ou une note midi.

2.6.2 Function Documentation

2.6.2.1 void jouerNote (const uint8_t & *midi_note*)

jouerNote permet de jouer une note midi de 45 a 81.

Parameters

midi_note	Note midi a jouer.

2.6.2.2 void jouerSon (const uint8_t & *v*)

Permet de faire jouer un son.

Parameters

v	Valeur de 0 a 255.

2.6.2.3 void silence ( )

silence permet d'arreter le piezo de jouer la note en cour.

2.6.2.4 void timer0_init (void)

timer0_init permet d'initialiser les bonnes valeurs de prescale et d'initiation du Timer0.

2.7 Moteur

Permet de la vitesse des deux moteurs.

Functions

- void activerMoteurs ()

  activerMoteurs, permet d'activer les moteurs en initialisant les registres du timer 1.
- void desactiverMoteur ()

  La fonction qui fait arreter le robot, en arretant le moteur droit et gauche.
- void ajusterPWM (const uint8_t &valeurRegistreComparaisonA, const uint8_t &valeurRegistreComparaisonB)

  ajusterPWM ajuste le rapport A/B du PWM en mettant les valeurs de comparaison dans les registres OCR1A et OCRB1.
- uint8_t convertirPourcentageEnInt (const uint8_t &pourcentage)

  convertirPourcentageEnInt convertit un pourcentage en valeur entiere pour generer le PWM correspondante.
- void tournerAGauche (const uint8_t &vitesseMoteurDroit)

  tournerAGauche permet de tourner le robot a gauche, on arrete le moteur gauche et on avance le moteur droit.
- void tournerADroite (const uint8_t &vitesseMoteurGauche)

  tournerADroite permet de tourner le robot a droite, on arrete le moteur droite et on avance le moteur gauche.
- void avancerRobot (const uint8_t &vitesseMoteurDroit, const uint8_t &vitesseMoteurGauche)

  avacerRobot permet d'avancer le robot, on avance le moteur droit et gauche en meme temps.
- void reculerRobot (const uint8_t &vitesseMoteurDroit, const uint8_t &vitesseMoteurGauche)

  reculerRobot permet de reculer le robot, on met en marche arriere le moteur droit et gauche en meme temps.
- void arreterRobot ()

  arreterRobot permet d'arreter le robot, en arretant le moteur droit et gauche.

2.7.1 Detailed Description

Permet de la vitesse des deux moteurs.

2.7.2 Function Documentation

2.7.2.1 void activerMoteurs ( )

activerMoteurs, permet d'activer les moteurs en initialisant les registres du timer 1.

2.7.2.2 void ajusterPWM (const uint8_t & *valeurRegistreComparaisonA*, const uint8_t & *valeurRegistreComparaisonB*)

ajusterPWM ajuste le rapport A/B du PWM en mettant les valeurs de comparaison dans les registres OCR1A et OCRB1.

Parameters

valeurRegistreComparaisonA	Valeur de comparaison mise dans le registre OCR1A.
valeurRegistreComparaisonB	Valeur de comparaison mise dans le registre OCR1B.

2.7.2.3 void arreterRobot ( )

arreterRobot permet d'arreter le robot, en arretant le moteur droit et gauche.

2.7.2.4 void avancerRobot (const uint8_t & *vitesseMoteurDroit,* const uint8_t & *vitesseMoteurGauche*)

avacerRobot permet d'avancer le robot, on avance le moteur droit et gauche en meme temps.

Parameters

vitesseMoteur-Droit	Permet de choisir la vitesse du moteur droit.
vitesseMoteur-Gauche	Permet de choisir la vitesse du moteur gauche.

2.7.2.5 uint8_t convertirPourcentageEnInt (const uint8_t & *pourcentage*)

convertirPourcentageEnInt convertit un pourcentage en valeur entiere pour generer le PWM correspondante.

Parameters

pourcentage	Valeur entre 0.0 et 100.0.

Returns

Retour la valeur en pourcentage.

2.7.2.6 void desactiverMoteur ( )

La fonction qui fait arreter le robot, en arretant le moteur droit et gauche.

2.7.2.7 void reculerRobot (const uint8_t & *vitesseMoteurDroit,* const uint8_t & *vitesseMoteurGauche*)

reculerRobot permet de reculer le robot, on met en marche arriere le moteur droit et gauche en meme temps.

Parameters

vitesseMoteur-Droit	Permet de choisir la vitesse du moteur droit.
vitesseMoteur-Gauche	Permet de choisir la vitesse du moteur gauche.

2.7.2.8 void tournerADroite (const uint8_t & *vitesseMoteurGauche*)

tournerADroite permet de tourner le robot a droite, on arrete le moteur droite et on avance le moteur gauche.

Parameters

vitesseMoteur-Gauche	Permet de choisir la vitesse pour tourner.

2.7.2.9 void tournerAGauche (const uint8_t & *vitesseMoteurDroit*)

tournerAGauche permet de tourner le robot a gauche, on arrete le moteur gauche et on avance le moteur droit.

Parameters

vitesseMoteur-Droit	Permet de choisir la vitesse pour tourner.

2.8 PhotoResistance

Capte les valeurs des photoresistances pour le robot souris .

Functions

- uint8_t luminositeGauche ()

  Fonction qui retourne la luminosite de gauche suite à une nouvelle lecture de la photoresistance de gauche (port #7).
- uint8_t luminositeDroite ()

  Fonction qui retourne la luminosite de droite suite à une nouvelle lecture de la photoresistance de droite (port #8).
- bool isIllumineGauche ()

  Fonction qui retourne l'état d'illumination de la photoresistance de gauche.
- bool isIllumineDroite ()

  Fonction qui retourne l'état d'illumination de la photoresistance de droite.
- void afficherPhotoresistances ()

  Fonction qui affiche les valeurs lues par les photoresistances au display branché au port C. À mettre dans une boucle si on veut afficher continuellement. La fonction comprend un délai de 400ms à la fin.

2.8.1 Detailed Description

Capte les valeurs des photoresistances pour le robot souris .

2.8.2 Function Documentation

2.8.2.1 void afficherPhotoresistances ( )

Fonction qui affiche les valeurs lues par les photoresistances au display branché au port C. À mettre dans une boucle si on veut afficher continuellement. La fonction comprend un délai de 400ms à la fin.

2.8.2.2 bool isIllumineDroite ( )

Fonction qui retourne l'état d'illumination de la photoresistance de droite.

Returns

Vrai lorsque la photoresistance de droite détecte une illumination .

2.8.2.3 bool isIllumineGauche ( )

Fonction qui retourne l'état d'illumination de la photoresistance de gauche.

Returns

Vrai lorsque la photoresistance de gauche détecte une illumination .

2.8.2.4 uint8_t luminositeDroite ( )

Fonction qui retourne la luminosite de droite suite à une nouvelle lecture de la photoresistance de droite (port #8).

Returns

La valeur de la luminosite de droite (0 - 255, sombre à clair).

2.8.2.5 uint8_t luminositeGauche ( )

Fonction qui retourne la luminosite de gauche suite à une nouvelle lecture de la photoresistance de gauche (port #7).

Returns

La valeur de la luminosite de gauche (0 - 255, sombre à clair) .

2.9 RobotUtils

Permet de gerer plusieurs structures pour le robot.

Enumerations

- enum Direction { direction_continue, direction_left, direction_right, direction_reculer }

  Enumeration pour les directions.
- enum Capteur { capteur_aucun = 255, capteur_droit = PINA5, capteur_centre = PINA3, capteur_gauche = PINA1 }

  Enumeration pour le capteur de ligne.
- enum Intersection {
  inter_none, inter_cross, inter_tBase, inter_tLeft,
  inter_tRight, inter_lLeft, inter_lRight }

  Enumeration de tous les intersections possibles.

2.9.1 Detailed Description

Permet de gerer plusieurs structures pour le robot.

2.9.2 Enumeration Type Documentation

2.9.2.1 enum Capteur

Enumeration pour le capteur de ligne.

Enumerator:

capteur_aucun

capteur_droit

capteur_centre

capteur_gauche

2.9.2.2 enum Direction

Enumeration pour les directions.

Enumerator:

direction_continue

direction_left

direction_right

direction_reculer

2.9.2.3 enum Intersection

Enumeration de tous les intersections possibles.

Enumerator:

inter_none

inter_cross

inter_tBase

inter_tLeft

inter_tRight

inter_lLeft

inter_lRight

Chapter 3

Class Documentation

3.1 LineFollower Class Reference

`#include <lineFollower.h>`

Public Member Functions

- LineFollower (void(*func)(LineFollower *follower, const Intersection &type))
- void followLine ()

  Permet de suivre la ligne et appel la fonction handelIntersection lorsque plus de deux sensors dont detectes.
- void turnLeft ()

  Permet de tourner a gauche.
- void turnRight ()

  Permet de tourner a droite.

Private Member Functions

- bool isCenter ()

  Permet de savoir si le capteur central est detectee.
- bool isLeft ()

  Permet de savoir si le capteur de gauche est detectee.
- bool isRight ()

  Permet de savoir si le capteur de droite est detectee.
- bool isOnlyCenterOn ()

  Permet de savoir si seulement le capteur central est detectee.
- bool isOnlyLeftOn ()

  Permet de savoir si seulement le capteur de gauche est detectee.
- bool isOnlyRightOn ()

  Permet de savoir si seulement le capteur de droite est detecteee.
- bool isNotOnLine ()

  Permet de savoir si le capteur de droite est detectee.
- bool isAllOnLine ()

  Permet de savoir si tous les capteurs sont detectes.
- bool isOnlyLeftOff ()

  Permet de savoir si tous les capteurs sont pas detectes.
- bool isOnlyRightOff ()

  Permet de savoir si seulement le capteur central est eteint.

- bool isMoreThanOneOn ()

  Permet de savoir si plus de deux capteurs sont allumes.
- bool isAtLeastOneOn ()

  Permet de savoir si au moin un capteur est allume.
- bool lookForCenterLineAfterIntersection (const Direction &direction)

  Permet de balayer, en tournant a gauche et a droite, pour detecter si le robot est toujours sur la ligne.
- void handleIntersection ()

  Permet de de choisir quelle intersection qui sera a gerer.
- void handleTLeftAndLRight ()
- void handleTRightAndLLeft ()
- void handleTBaseAndCross ()
- void keepGoingAlittleBit ()

Private Attributes

- Direction m_last_straight_line_direction
- void(∗ m_fct_ptr)(LineFollower ∗follower, const Intersection &type)

  Variable pour se rappeler de la fonction a appeler.

Static Private Attributes

- static const uint8_t c_moteur_speed = 160
- static const uint8_t c_slope_speed = 100

3.1.1 Detailed Description

Permet la gestion des lignes et intersections. Lorsque le capteur tttraverse une intersection, une fonction passee en parametre par pointeur est appelee. Cette fonction, definie par l'utilisateur, permet de choisir la direction qui le robot prendra. Si les fonctions pour tourner ne sont pas appelees, le robot va continuer a avancer. La classe permet egalement de tourner automatiquement au intersection aux coins.

Parameters

lcd	Classe LCM pour utiliser la fonction write sur le bon port.
value	Valeur a etre ecrite.
value_number	Numero de la valeur soit (0, 1, 2, 3) pour V0 a V3.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 LineFollower::LineFollower (void(∗)(LineFollower ∗follower, const Intersection &type) *func*)

3.1.3 Member Function Documentation

3.1.3.1 void LineFollower::followLine ( )

Permet de suivre la ligne et appel la fonction handelIntersection lorsque plus de deux sensors dont detectes.

3.1.3.2 void LineFollower::handleIntersection ( ) [private]

Permet de de choisir quelle intersection qui sera a gerer.

3.1.3.3 void LineFollower::handleTBaseAndCross ( ) [private]

3.1.3.4 void LineFollower::handleTLeftAndLRight ( ) [private]

3.1.3.5 void LineFollower::handleTRightAndLLeft ( ) [private]

3.1.3.6 bool LineFollower::isAllOnLine ( ) [private]

Permet de savoir si tous les capteurs sont detectes.

3.1.3.7 bool LineFollower::isAtLeastOneOn ( ) [private]

Permet de savoir si au moin un capteur est allume.

3.1.3.8 bool LineFollower::isCenter ( ) [private]

Permet de savoir si le capteur central est detectee.

3.1.3.9 bool LineFollower::isLeft ( ) [private]

Permet de savoir si le capteur de gauche est detectee.

3.1.3.10 bool LineFollower::isMoreThanOneOn ( ) [private]

Permet de savoir si plus de deux capteurs sont allumes.

3.1.3.11 bool LineFollower::isNotOnLine ( ) [private]

Permet de savoir si le capteur de droite est detectee.

3.1.3.12 bool LineFollower::isOnlyCenterOn ( ) [private]

Permet de savoir si seulement le capteur central est detectee.

3.1.3.13 bool LineFollower::isOnlyLeftOff ( ) [private]

Permet de savoir si tous les capteurs sont pas detectes.

3.1.3.14 bool LineFollower::isOnlyLeftOn ( ) [private]

Permet de savoir si seulement le capteur de gauche est detectee.

3.1.3.15 bool LineFollower::isOnlyRightOff ( ) [private]

Permet de savoir si seulement le capteur central est eteint.

3.1.3.16 bool LineFollower::isOnlyRightOn ( ) [private]

Permet de savoir si seulement le capteur de droite est detecteee.

3.1.3.17 bool LineFollower::isRight ( ) [private]

Permet de savoir si le capteur de droite est detectee.

3.1.3.18 void LineFollower::keepGoingAlittleBit ( ) [private]

3.1.3.19 bool LineFollower::lookForCenterLineAfterIntersection (const Direction & *direction*) [private]

Permet de balayer, en tournant a gauche et a droite, pour detecter si le robot est toujours sur la ligne.

3.1.3.20 void LineFollower::turnLeft ( )

Permet de tourner a gauche.

3.1.3.21 void LineFollower::turnRight ( )

Permet de tourner a droite.

3.1.4 Member Data Documentation

3.1.4.1 const uint8_t LineFollower::c_moteur_speed = 160 [static], [private]

3.1.4.2 const uint8_t LineFollower::c_slope_speed = 100 [static], [private]

3.1.4.3 void(* LineFollower::m_fct_ptr)(LineFollower *follower, const Intersection &type) [private]

Variable pour se rappeler de la fonction a appeler.

3.1.4.4 Direction LineFollower::m_last_straight_line_direction [private]

The documentation for this class was generated from the following files:

- lineFollower.h
- lineFollower.cpp

Index