My Project

Generated by Doxygen 1.8.16

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 camera Namespace Reference

5.1.1 Detailed Description

image analysis

5.2 Camera Namespace Reference

Classes

- class Camera

  take a picture and anlyse it to detect a dice

5.3 main Namespace Reference

Classes

- class RobotControl_Thread

  Class RobotControl_Thread update postion of the robot.

Functions

- def getData ()

  launch robotController Thread
- def stateMachine (ev=int)

  stateMachine manage the state of the soft : init, running, stop

Variables

- int state = STATE_INIT

  actual state for the state machine
- int oldState = STATE_INIT

  old state for the state machine
- theRobotController = RobotControl()

  class RobotControl object
- theCamera = Camera()

  class Camera object
- t = Timer(0.1,getData)

  timer to launch thread periodically

5.3.1 Detailed Description

main of the program

5.3.2 Function Documentation

5.3.2.1 getData()

```
def main.getData ( )
```

launch robotController Thread

5.3.2.2 stateMachine()

```
def main.stateMachine (
              ev = int )
```

stateMachine manage the state of the soft : init, running, stop

Parameters

ev	event to trigger the state machine

5.3.3 Variable Documentation

5.3.3.1 oldState

```
int main.oldState = STATE_INIT
```

old state for the state machine

5.3.3.2 state

```
int main.state = STATE_INIT
```

actual state for the state machine

5.3.3.3 t

```
main.t = Timer(0.1,getData)
```

timer to launch thread periodically

5.3.3.4 theCamera

```
main.theCamera = Camera()
```

class Camera object

5.3.3.5 theRobotController

```
main.theRobotController = RobotControl()
```

class RobotControl object

5.4 RobotControl Namespace Reference

Classes

- class RobotControl

  Control the robot.

5.4.1 Detailed Description

control the robot

Chapter 6

Class Documentation

6.1 Camera.Camera Class Reference

take a picture and anlyse it to detect a dice

Public Member Functions

- def __init__ (self)

    the constructor
- def initRelation (self, robotController)

    intialise relation
- def capture (self)

    capture an image
- def cameraDetectionDice (self)

    manage the dice detection then trigger the state machine of class RobotControl depending on the dice number
- def detectNumberOnDice (self, image)

    detect the number on a dice
- def midpoint (self, ptA, ptB)

    calculate the mid point between 2 points
- def foundDice (self, imagePath, width_object)

    search a dice in an image

Public Attributes

- deltaX_m

    delta x of the object in meter from the center of the camera
- deltaY_m

    delta y of the object in meter from the center of the camera
- angleRot

    orientation the object in radian
- pixelsPerMeter

    pixels per meter ration
- imgCrop

    image of the dice
- robotController

    robotController object
- camera

    camera object

6.1.1 Detailed Description

take a picture and anlyse it to detect a dice

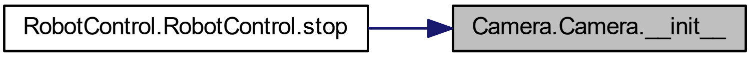
6.1.2 Constructor & Destructor Documentation

6.1.2.1 __init__()

```
def Camera.Camera.__init__ (
              self )
```

the constructor

Here is the caller graph for this function:



6.1.3 Member Function Documentation

6.1.3.1 cameraDetectionDice()

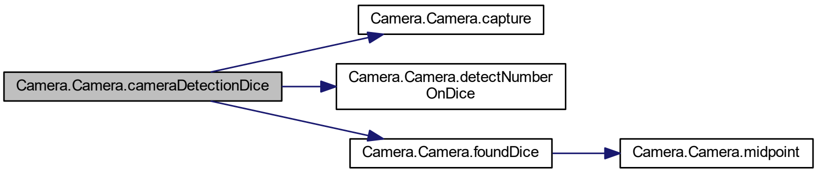
```
def Camera.Camera.cameraDetectionDice (
              self )
```

manage the dice detection then trigger the state machine of class RobotControl depending on the dice number

Parameters

self	The object pointer.

Here is the call graph for this function:


Camera.Camera.capture
Camera.Camera.cameraDetectionDice
Camera.Camera.detectNumber OnDice
Camera.Camera.foundDice
Camera.Camera.midpoint

6.1.3.2 capture()

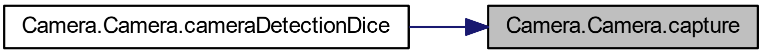
```
def Camera.Camera.capture (
              self )
```

capture an image

Parameters

self	The object pointer.

Here is the caller graph for this function:



6.1.3.3 detectNumberOnDice()

```
def Camera.Camera.detectNumberOnDice (
              self,
              image )
```

detect the number on a dice

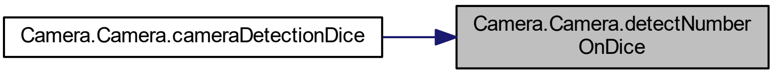
Parameters

self	The object pointer.
image	image of the dice

Returns

the dice number

Here is the caller graph for this function:


Camera.Camera.cameraDetectionDice
Camera.Camera.detectNumber OnDice

6.1.3.4 foundDice()

```
def Camera.Camera.foundDice (
              self,
              imagePath,
              width_object )
```

search a dice in an image

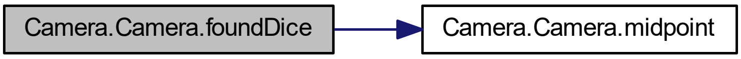
Parameters

self	The object pointer.
imagePath	path to the image
width_object	width in mm of the object

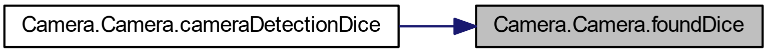
Returns

an image of the dice

Here is the call graph for this function:



Here is the caller graph for this function:



6.1.3.5 initRelation()

```
def Camera.Camera.initRelation (
              self,
              robotController )
```

intialise relation

Parameters

self	The object pointer.
self	The robot controller object

6.1.3.6 midpoint()

```
def Camera.Camera.midpoint (
              self,
              ptA,
              ptB )
```

calculate the mid point between 2 points

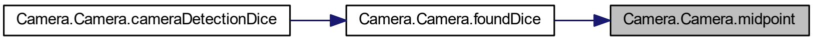
Parameters

self	The object pointer.
ptA	point 1
ptB	point 2

Returns

the middle point

Here is the caller graph for this function:



6.1.4 Member Data Documentation

6.1.4.1 angleRot

```
Camera.Camera.angleRot
```

orientation the object in radian

6.1.4.2 camera

```
Camera.Camera.camera
```

camera object

6.1.4.3 deltaX_m

```
Camera.Camera.deltaX_m
```

delta x of the object in meter from the center of the camera

6.1.4.4 deltaY_m

```
Camera.Camera.deltaY_m
```

delta y of the object in meter from the center of the camera

6.1.4.5 imgCrop

```
Camera.Camera.imgCrop
```

image of the dice

6.1.4.6 pixelsPerMeter

```
Camera.Camera.pixelsPerMeter
```

pixels per meter ration

6.1.4.7 robotController

```
Camera.Camera.robotController
```

robotController object

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

- C:/Users/rebor/Documents/GitHub/PGA/Code/Camera.py

6.2 RobotControl.RobotControl Class Reference

Control the robot.

Public Member Functions

- def __init__ (self)

  constructor
- def initRelations (self, theCamera)

  get camera object
- def calibrate (self)

  Put the robot in original state.
- def updateCurrentPosition (self)

  update robot position get position X,Y,Z and orientation of tool center point
- def moveToPosition (self, x=float, y=float, z=float, rz=float)

  move the robot to a position XYZ with angle rz
- def setObjectPosition (self, dx=float, dy=float, rz=float)

  set the object found postion
- def statePliers (self)

  check the state of the pliers
- def adjustPliers (self, pliersState)

  open or close the pliers
- def master (self, event)

  State machine who manage the soft as follow : Step 1 : Move the pliers Step 2 : Search the dice Step 3 : dice found --> go step 4, else step 1 Step 4 : Grab the dice Step 5 : Launch the dice and go back to step 1 Stop if the dice is 6.
- def stop (self)

  this function stop the robot and put him in original state

Public Attributes

- angularvelocity

  angular velocity of the robot
- angularacceleration

  angular acceleration of the robot
- linearvelocity

  linear velocity of the robot
- linearacceleration

  linear acceleration of the robot
- posx

  tool center point position x
- posy

  tool center point position y
- posz

  tool center point position z
- rx

  tool center point orientation rx
- ry

  tool center point orientation ry
- rz

  tool center point orientation rz
- object_posX

  x postion of the object to catch
- object_posY

  y postion of the object to catch
- object_Rz
- takeOrRelease
- ZeroReached
- MaxReached
- evZone
- lastZone
- BorderReached
- xSearch

  position x to search the object
- ySearch

  position y to search the object
- zSearch

  position z to search the object
- host

  IP Address of the robot.
- state

  state of the state machine
- oldState

  oldstate of the state machine
- theCamera

  camera object
- object_posZ

6.2.1 Detailed Description

Control the robot.

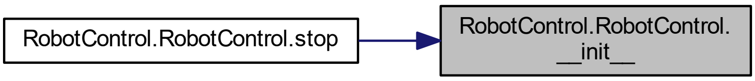
6.2.2 Constructor & Destructor Documentation

6.2.2.1 __init__()

```
def RobotControl.RobotControl.__init__ (
              self )
```

constructor

Here is the caller graph for this function:



6.2.3 Member Function Documentation

6.2.3.1 adjustPliers()

```
def RobotControl.RobotControl.adjustPliers (
              self,
              pliersState )
```

open or close the pliers

Parameters

pliersState	True->Open, False->Close
self	The object pointer.

6.2.3.2 calibrate()

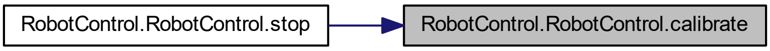
```
def RobotControl.RobotControl.calibrate (
              self )
```

Put the robot in original state.

Parameters

self	The object pointer.

Here is the caller graph for this function:


RobotControl.RobotControl.stop
RobotControl.RobotControl.calibrate

6.2.3.3 initRelations()

```
def RobotControl.RobotControl.initRelations (
              self,
              theCamera )
```

get camera object

Parameters

self	The object pointer.
theCamera	the camera object.

6.2.3.4 master()

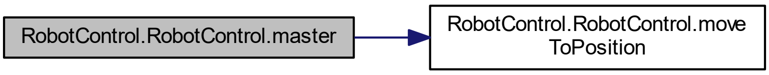
```
def RobotControl.RobotControl.master (
              self,
              event )
```

State machine who manage the soft as follow : Step 1 : Move the pliers Step 2 : Search the dice Step 3 : dice found --> go step 4, else step 1 Step 4 : Grab the dice Step 5 : Launch the dice and go back to step 1 Stop if the dice is 6.

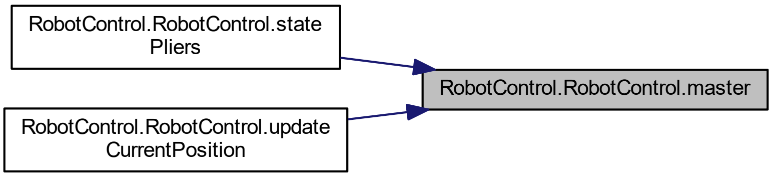
Parameters

self	The object pointer.
event	event to trigger state machine.

Here is the call graph for this function:


RobotControl.RobotControl.master
RobotControl.RobotControl.move ToPosition

Here is the caller graph for this function:


RobotControl.RobotControl.state Pliers
RobotControl.RobotControl.update CurrentPosition
RobotControl.RobotControl.master

6.2.3.5 moveToPosition()

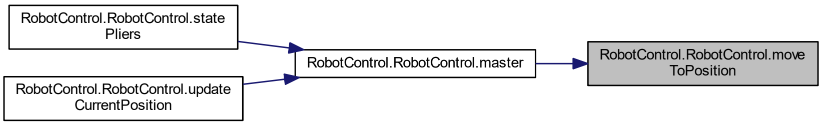
```
def RobotControl.RobotControl.moveToPosition (
              self,
              x = float,
              y = float,
              z = float,
              rz = float )
```

move the robot to a position XYZ with angle rz

Parameters

x	: Position X of the Tool Center Point
y	: Position Y of the Tool Center Point
z	: Position Z of the Tool Center Point
rz	: orientation of the object
self	The object pointer.

Here is the caller graph for this function:


RobotControl.RobotControl.state Pliers
RobotControl.RobotControl.update CurrentPosition
RobotControl.RobotControl.master
RobotControl.RobotControl.move ToPosition

6.2.3.6 setObjectPosition()

```
def RobotControl.RobotControl.setObjectPosition (
              self,
              dx = float,
              dy = float,
              rz = float )
```

set the object found postion

Parameters

dx	: Delta X of the object from the center of the camera
dy	: Delta Y of the object from the center of the camera
rz	: orientation of the object
self	The object pointer.

6.2.3.7 statePliers()

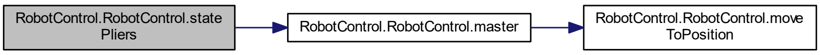
```
def RobotControl.RobotControl.statePliers (
              self )
```

check the state of the pliers

Parameters

self	The object pointer.

Here is the call graph for this function:


RobotControl.RobotControl.state Pliers
RobotControl.RobotControl.master
RobotControl.RobotControl.move ToPosition

6.2.3.8 stop()

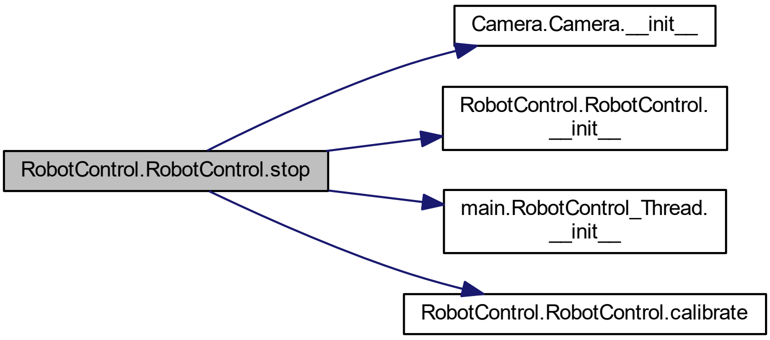
```
def RobotControl.RobotControl.stop (
              self )
```

this function stop the robot and put him in original state

Parameters

self	The object pointer.

Here is the call graph for this function:


Camera.Camera.__init__
RobotControl.RobotControl.__init__
RobotControl.RobotControl.stop
main.RobotControl_Thread.__init__
RobotControl.RobotControl.calibrate

6.2.3.9 updateCurrentPosition()

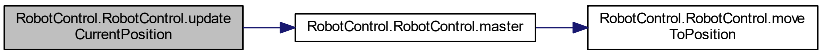
```
def RobotControl.RobotControl.updateCurrentPosition (
              self )
```

update robot position get position X,Y,Z and orientation of tool center point

Parameters

self	The object pointer.

Here is the call graph for this function:


RobotControl.RobotControl.update CurrentPosition
RobotControl.RobotControl.master
RobotControl.RobotControl.move ToPosition

6.2.4 Member Data Documentation

6.2.4.1 angularacceleration

```
RobotControl.RobotControl.angularacceleration
```

angular acceleration of the robot

6.2.4.2 angularvelocity

```
RobotControl.RobotControl.angularvelocity
```

angular velocity of the robot

6.2.4.3 BorderReached

```
RobotControl.RobotControl.BorderReached
```

6.2.4.4 evZone

```
RobotControl.RobotControl.evZone
```

6.2.4.5 host

```
RobotControl.RobotControl.host
```

IP Address of the robot.

6.2.4.6 lastZone

```
RobotControl.RobotControl.lastZone
```

6.2.4.7 linearacceleration

```
RobotControl.RobotControl.linearacceleration
```

linear acceleration of the robot

6.2.4.8 linearvelocity

```
RobotControl.RobotControl.linearvelocity
```

linear velocity of the robot

6.2.4.9 MaxReached

```
RobotControl.RobotControl.MaxReached
```

6.2.4.10 object_posX

```
RobotControl.RobotControl.object_posX
```

x postion of the object to catch

6.2.4.11 object_posY

```
RobotControl.RobotControl.object_posY
```

y postion of the object to catch

6.2.4.12 object_posZ

```
RobotControl.RobotControl.object_posZ
```

6.2.4.13 object_Rz

```
RobotControl.RobotControl.object_Rz
```

6.2.4.14 oldState

```
RobotControl.RobotControl.oldState
```

oldstate of the state machine

6.2.4.15 posx

```
RobotControl.RobotControl.posx
```

tool center point position x

6.2.4.16 posy

```
RobotControl.RobotControl.posy
```

tool center point position y

6.2.4.17 posz

```
RobotControl.RobotControl.posz
```

tool center point position z

6.2.4.18 rx

```
RobotControl.RobotControl.rx
```

tool center point orientation rx

6.2.4.19 ry

```
RobotControl.RobotControl.ry
```

tool center point orientation ry

6.2.4.20 rz

```
RobotControl.RobotControl.rz
```

tool center point orientation rz

6.2.4.21 state

```
RobotControl.RobotControl.state
```

state of the state machine

6.2.4.22 takeOrRelease

```
RobotControl.RobotControl.takeOrRelease
```

6.2.4.23 theCamera

```
RobotControl.RobotControl.theCamera
```

camera object

6.2.4.24 xSearch

```
RobotControl.RobotControl.xSearch
```

position x to search the object

6.2.4.25 ySearch

```
RobotControl.RobotControl.ySearch
```

position y to search the object

6.2.4.26 ZeroReached

```
RobotControl.RobotControl.ZeroReached
```

6.2.4.27 zSearch

```
RobotControl.RobotControl.zSearch
```

position z to search the object

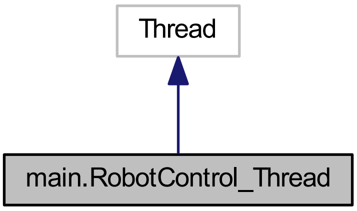
The documentation for this class was generated from the following file:

- C:/Users/rebor/Documents/GitHub/PGA/Code/RobotControl.py

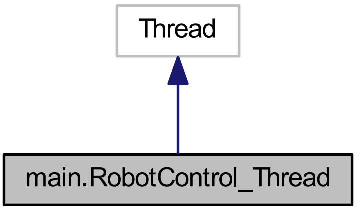
6.3 main.RobotControl_Thread Class Reference

Class RobotControl_Thread update postion of the robot.

Inheritance diagram for main.RobotControl_Thread:



Collaboration diagram for main.RobotControl_Thread:



Public Member Functions

- def __init__ (self)

  the constructor
- def run (self)

  get position of the robot

6.3.1 Detailed Description

Class RobotControl_Thread update postion of the robot.

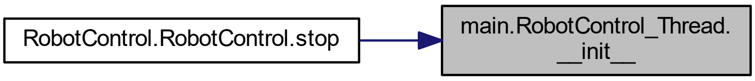
6.3.2 Constructor & Destructor Documentation

6.3.2.1 __init__()

```
def main.RobotControl_Thread.__init__ (
              self )
```

the constructor

Here is the caller graph for this function:



6.3.3 Member Function Documentation

6.3.3.1 run()

```
def main.RobotControl_Thread.run (
              self )
```

get position of the robot

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

- C:/Users/rebor/Documents/GitHub/PGA/Code/main.py

Chapter 7

File Documentation

7.1 C:/Users/rebor/Documents/GitHub/PGA/Code/Camera.py File Reference

Classes

- class Camera.Camera

  take a picture and anlyse it to detect a dice

Namespaces

- Camera
- camera

7.2 C:/Users/rebor/Documents/GitHub/PGA/Code/main.py File Reference

Classes

- class main.RobotControl_Thread

  Class RobotControl_Thread update postion of the robot.

Namespaces

- main

Functions

- def main.getData ()

  launch robotController Thread
- def main.stateMachine (ev=int)

  stateMachine manage the state of the soft : init, running, stop

Variables

- int main.state = STATE_INIT

    actual state for the state machine
- int main.oldState = STATE_INIT

    old state for the state machine
- main.theRobotController = RobotControl()

    class RobotControl object
- main.theCamera = Camera()

    class Camera object
- main.t = Timer(0.1,getData)

    timer to launch thread periodically

7.3 C:/Users/rebor/Documents/GitHub/PGA/Code/RobotControl.py File Reference

Classes

- class RobotControl.RobotControl

    Control the robot.

Namespaces

- RobotControl