**YeastRobot – function definitions**

########### preliminaries ##########

## likely something like this at the start of each program

import sys

sys.path.append('/path/to/application/app/folder')

from RobotControl import \*

#################################

### Define Deck Layout

#################################

deck="""\

DW96P SW24P SW24P SW24P SW24P

DW96P SW24P SW24P SW24P SW24P

DW96P SW24P SW24P SW24P SW24P

DW96P SW24P SW24P SW24P SW24P

"""

# 2 3 4 5 6

# note the 1st user defined column is "2" not zero or one, since tips are at 0 & 1

##################################

## tip positions are numbered 1 through 24, 1 is Box A UL offset;

## 2 AUR; 3 ALL; 4 ALR; 5 BUL, …, 24 FLR

## Box A,B,C,D,E,F are at positions {0,0}, {0,1}, {1,0},{1,1}, etc. respectively

## if current position reaches “25” the program moved the head to the

## right and asks for new tip boxes.

CurrentTipPosition = 25

OffsetDict={1: 'UL', 2: 'UR', 3: 'LL', 4: 'LR'}

DefineDeck(deck)

## this function asks the user to OK the current deck

printDeck()

InitializeRobot()

CurrentTipPosition = retrieveTips(CurrentTipPosition)

## .... program here ...

fast\_home\_velmex()

ShutDownRobot()

########### end preliminaries ##########

retrieveTips(CurrentTipPosition)

description:

picks up the next tip

arguments:

CurrentTipPosition = tip to get min=1, max=24

returns:

next tip position

position(row, col, offset="UL")

description:

moves the head to a position on the deck

arguments:

row: 0 ... nrows defined in "deck"

col: 0 ... ncol defined in "deck"

offset: "UL","UR","LL","LR"

default is UL

UL is center of well for 24 well plates

aspirate(volumn, percent\_down, percent\_speed)

description:

aspirates liquid from plate. Goes straight down from last

position, and returns to "safe\_height" when done

arguments:

volumn: desired volume in microliters

percent\_down: percent of the way from "surf\_height" to "max\_height"

to pipette from. 100 is bottom of plate.

percent\_speed: percent speed to aspirate at

moveAspirate(volumn, percent\_start, percent\_finish, percent\_speed)

description:

aspirates while moving, useful in narrow well plates

arguments:

volumn: desired volume in microliters

percent\_start: percent of the way from "surf\_height" to "max\_heigth"

to start aspirating

percent\_end: percent of the way to stop at

percent\_speed: percent speed

dispense(volumn, percent\_down, percent\_speed, blowout = 'N')

description:

dispense liquid to plate. Goes straight down from last

position, and returns to "safe\_height" when done

arguments:

volumn: desired volume in microliters

percent\_down: percent of the way from "surf\_height" to "max\_height"

to pipette from. 100 is bottom of plate.

percent\_speed: percent speed to aspirate at

blow\_out: after dispensing blow out any remaining liquid with air

default is "N" = no

moveDispense(volumn, percent\_start, percent\_speed, blowout = 'N')

description:

dispenses while moving, useful in narrow well plates

arguments:

volumn: desired volume in microliters

percent\_start: percent of the way from "surf\_height" to "max\_heigth"

to start dispensing (stops at "surf\_height")

percent\_speed: percent speed

blow\_out: after dispensing blow out any remaining liquid with air

default is "N" = no

liquidDisposal()

description:

goes to liquid disposal station (at 4,0) and gets rid of

any liquid in tips

disposeTips()

description:

goes to tip disposal station (at 4,1) and ejects tips

enterToContinue()

description:

this can be useful if you need a user pause in a program

mix(volumn, percent\_down, percent\_speed)

still not rewritten

checkAlignment()

description:

checks X&Y alignment at some "position". Goes to "surfaceDepth"

and waits for user. Goes to "maxDepth" and waits for user. Returns

to "safeDepth". You can run this from the command line combined

with a sequence of "position" commands to check aligments

newplate(row=0, col=2)

description:

this command would follow a "position". It is usefully for quickly

defining the "offsets" for a new plate.

It takes the head to the

magical "zero, zero" position for a plate at the safeheight. It then

ask the user for a "number of mm to move down". You guess a distance

and the Z axis moves. The positive goes down, negative up. Once you

are happy with the height enter zero and the function gives you the

current height in steps and asks for an X offset. You can now adjust

X and get the steps returned, then Y and get the steps returned, and

finally maxDepth. It is pretty important the maxdepth of a plate

really is maxDepth = the point at which the tips almost touch the

bottom.

If you run this function at position {0,2} and define the centers

of the wells of a plate (and its heights), and then enter this

information in the "CellClass" file. And then you rerun the exercise

for a new plate position (e.g., {0,3}) -- if the well centers are

NOT CORRECT the problem is you magical "zero, zero" positions. The

offsets for a plate are CONSTANT from position to position.

So in theory you should only need to run this program at {0,2} and

check all the other positons using "checkAlignment".