AutoGrind User Manual

For

TOSOH

TOSOH QUARTZ, INC.

Olympus CONTROLS

Where Innovation Meets Automation®

Author: Ned Lecky
nlecky@olympus-controls.com

Revision 1.0
May 30, 2022

Contents

Overview

This is a first draft that covers the main features of the AutoGrind software.

There are four Operation Tabs: **Run**, **Program**, **Setup**, and **Log**. These screens are described below.

There are three User Modes. The user mode is selected using the **User** field in the upper-right corner of the Run tab

1. **Operator** Mode can only load and run existing recipes.
2. **Editor** Mode is Operator plus the user can create and edit recipes.
3. **Engineering** Mode is Editor plus access to all setup and configuration functions.

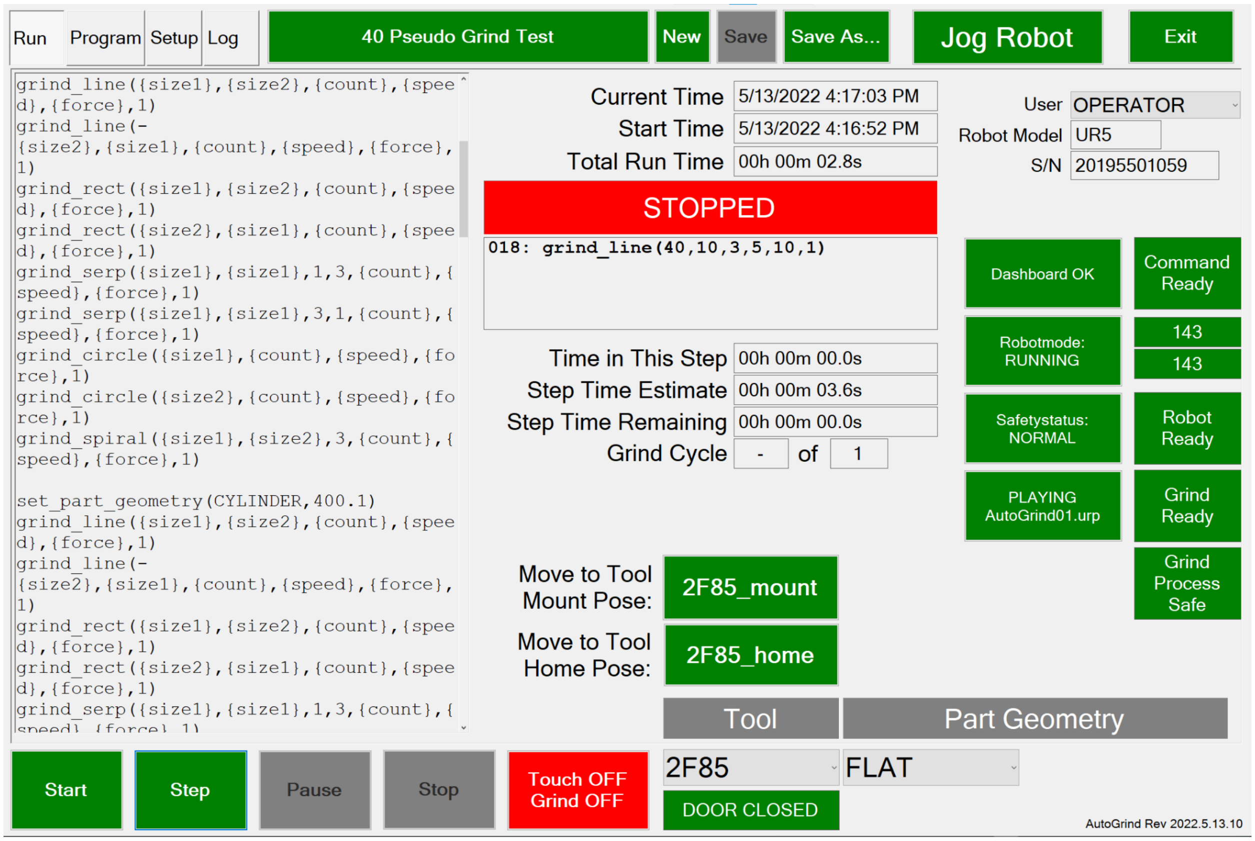
Entering Operator or Engineering mode requires a fixed password. By default, these are 9 and 99, respectively. They are not currently user settable.

Many functions can be manually activated with buttons or automatically activated with recipe commands. The convention in this manual is the used boldface for buttons and italics for recipe commands, as in **This Is a Button** and *this_is_a_recipe_function(param1)*.

System Tabs

Run Tab

The **Run Tab** is where the bulk of program execution will typically be observed.



```
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(CYLINDER,400.1)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
```

Current Time 5/13/2022 4:17:03 PM
Start Time 5/13/2022 4:16:52 PM
Total Run Time 00h 00m 02.8s

User OPERATOR
Robot Model UR5
S/N 20195501059

```
018: grind_line(40,10,3,5,10,1)
```

Time in This Step 00h 00m 00.0s
Step Time Estimate 00h 00m 03.6s
Step Time Remaining 00h 00m 00.0s
Grind Cycle - of 1

143
143

Tool: 2F85
Part Geometry: FLAT

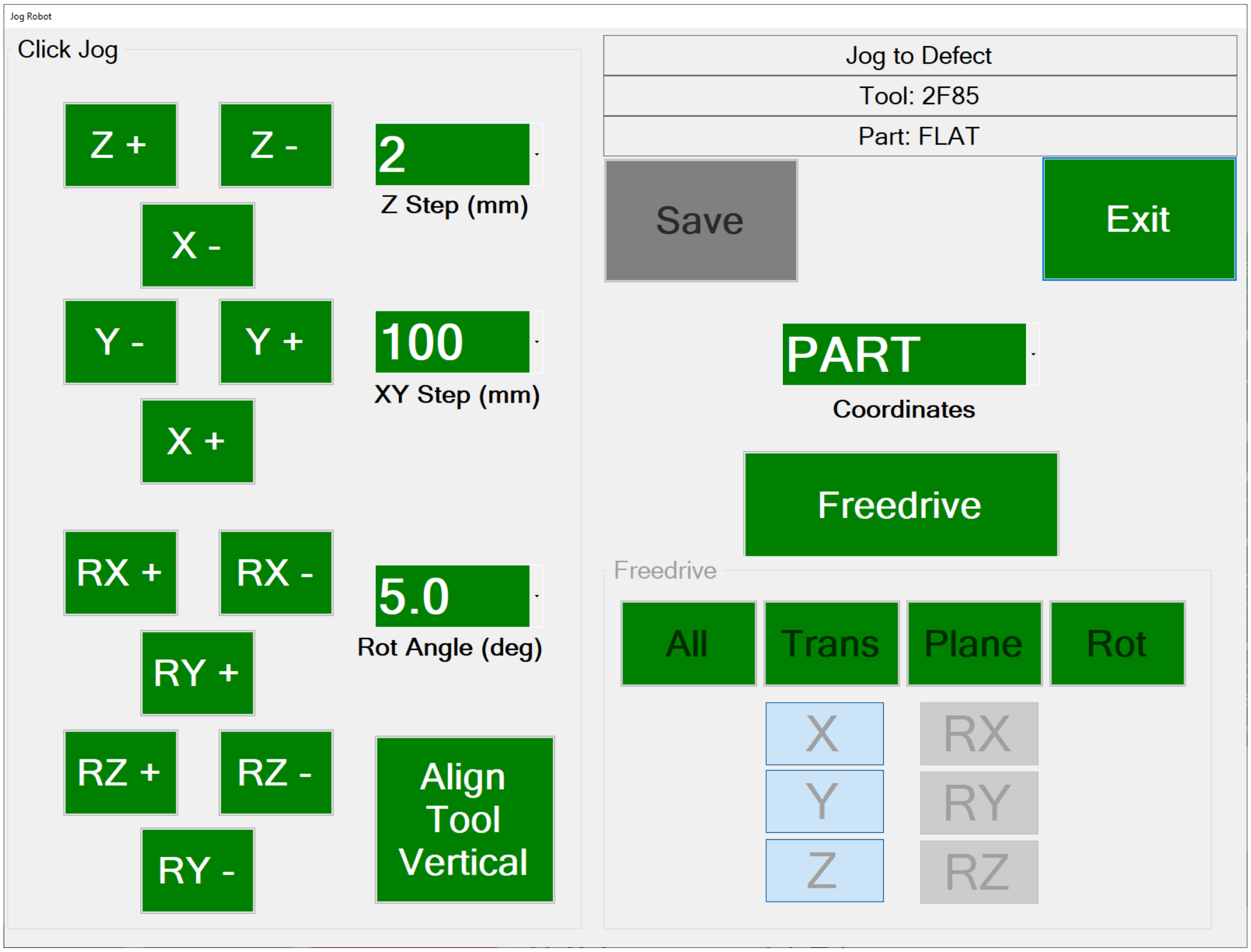
Tools are selected in the **Tool Dropdown.**
Part geometry in specified in **Part Geometry Dropdown:** FLAT, CYLINDER, and SPHERE
Tools have mount and home positions that can be moved into with the **toolname_mount** and **toolname_home** buttons.
A recipe is loaded using the **Recipe Name** button next to the **Log** tab.
Recipe file operations in addition to **Load** are **New, Save, Save As.**
Set the User in the **User Dropdown.**
Set the grinding mode with the **Touch/Grind button** which cycles through **No contact, Touch Only, and Touch+Grind**
Protective Stops: These show up in (and may be cleared with) the SafetyStatus button
Door Status is monitored (IO is configured in the **Setup Tab**). Door Open is treated like **Pause**.
Running a Recipe behaves as expected:

Start, **Stop**, **Step**, and **Pause / Continue**

Jog Robot is used to jog or freedrive to a defect. Jogging is described on the next page.

Robot Jogging in AutoGrind

Jogging opens a separate screen. Jogging can be done in **BASE** or **TOOL** coordinates, or relative to a **PART**. The buttons move the robot by the specified increment in Z, XY, or rotation. Holding a button down (mouse) or double-tapping and holding (tablet) makes the move repeat.



Jog Robot

Tool: 2F85

Part: FLAT

2 Z Step (mm)

100 XY Step (mm)

5.0 Rot Angle (deg)

PART Coordinates

When jogging in **PART** mode, if a cylindrical or spherical geometry is selected, the tool will rotate around the center of the part instead of around the tool tip. This can be convenient for manually jogging to a defect using the touch screen instead of freedrive

Freedrive is supported in a manner identical to on the UR pendant. The X, Y, X, RX, RY and RZ buttons may be used to enable or disable freedrive in any desired axis. All, Trans, Plane, and Rot select pre-defined subsets of axes as on the UR.

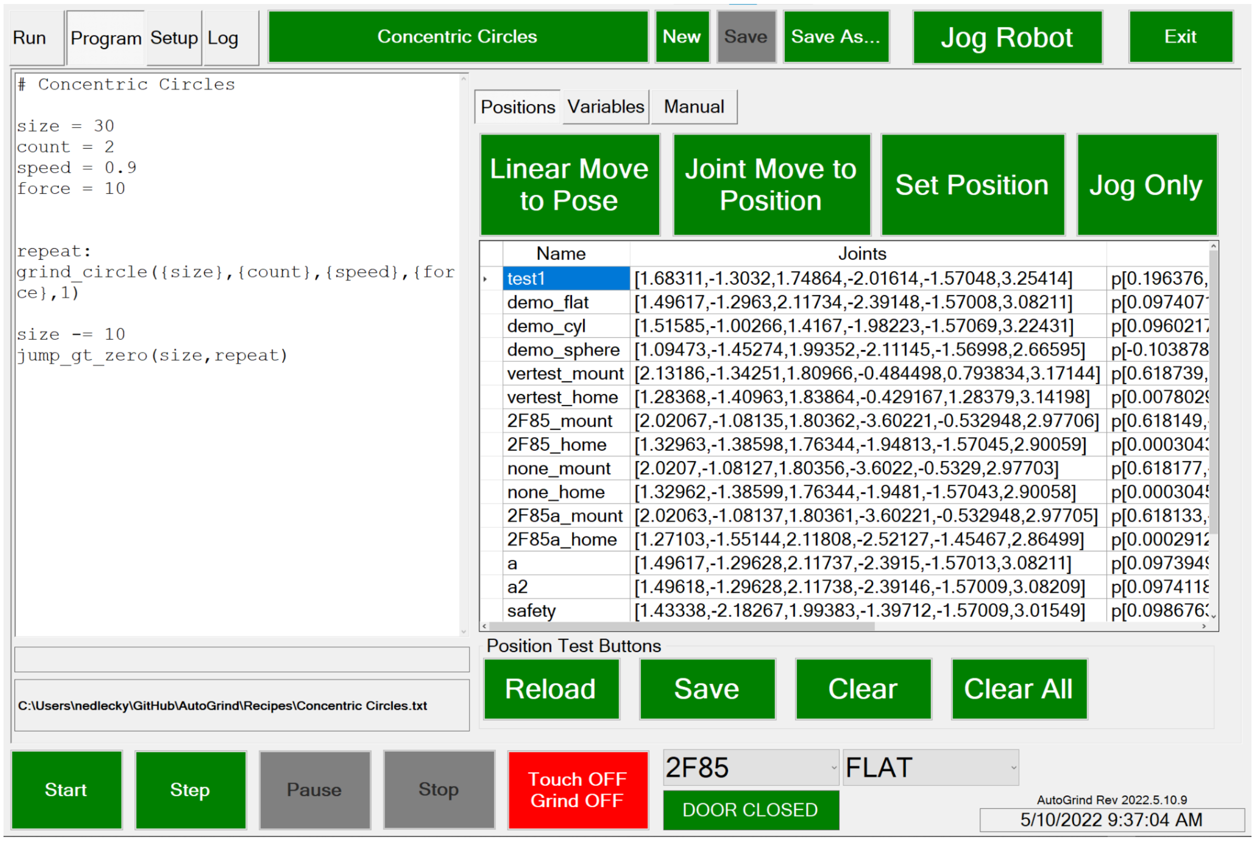
Coordinate systems may be changed during freedrive and the tool will allow motion relativ to the world, the tool, or the center of the part of part geometry is cylinder or sphere.
Press the Freedrive button again to turn freedrive mode off. Saving or exiting the dialog will also turn off freedrive.

Program Tab

The **Program Tab** has three sub pages: **Positions**, for teaching and manually moving to fixed positions, **Variables**, for monitoring or changing AutoGrind variables, and **Manual** which provides access to documentation on the recipe commands.

Program Tab - Positions

Below is the **Program Tab** when the Positions Subtab is selected.



```
# Concentric Circles

size = 30
count = 2
speed = 0.9
force = 10


repeat:
grind_circle({size},{count},{speed},{for
ce},1)

size -= 10
jump_gt_zero(size,repeat)
```

C:\Users\nedlecky\GitHub\AutoGrind\Recipes\Concentric Circles.txt

	Name	Joints	
▸	test1	[1.68311,-1.3032,1.74864,-2.01614,-1.57048,3.25414]	p[0.196376,
	demo_flat	[1.49617,-1.2963,2.11734,-2.39148,-1.57008,3.08211]	p[0.097407
	demo_cyl	[1.51585,-1.00266,1.4167,-1.98223,-1.57069,3.22431]	p[0.096021
	demo_sphere	[1.09473,-1.45274,1.99352,-2.11145,-1.56998,2.66595]	p[-0.103878
	vertest_mount	[2.13186,-1.34251,1.80966,-0.484498,0.793834,3.17144]	p[0.618739,
	vertest_home	[1.28368,-1.40963,1.83864,-0.429167,1.28379,3.14198]	p[0.007802
	2F85_mount	[2.02067,-1.08135,1.80362,-3.60221,-0.532948,2.97706]	p[0.618149,
	2F85_home	[1.32963,-1.38598,1.76344,-1.94813,-1.57045,2.90059]	p[0.000304
	none_mount	[2.0207,-1.08127,1.80356,-3.6022,-0.5329,2.97703]	p[0.618177,
	none_home	[1.32962,-1.38599,1.76344,-1.9481,-1.57043,2.90058]	p[0.000304
	2F85a_mount	[2.02063,-1.08137,1.80361,-3.60221,-0.532948,2.97705]	p[0.618133,
	2F85a_home	[1.27103,-1.55144,2.11808,-2.52127,-1.45467,2.86499]	p[0.000291
	a	[1.49617,-1.29628,2.11737,-2.3915,-1.57013,3.08211]	p[0.097394
	a2	[1.49618,-1.29628,2.11738,-2.39146,-1.57009,3.08209]	p[0.097411
	safety	[1.43338,-2.18267,1.99383,-1.39712,-1.57009,3.01549]	p[0.098676

AutoGrind Rev 2022.5.10.9
5/10/2022 9:37:04 AM

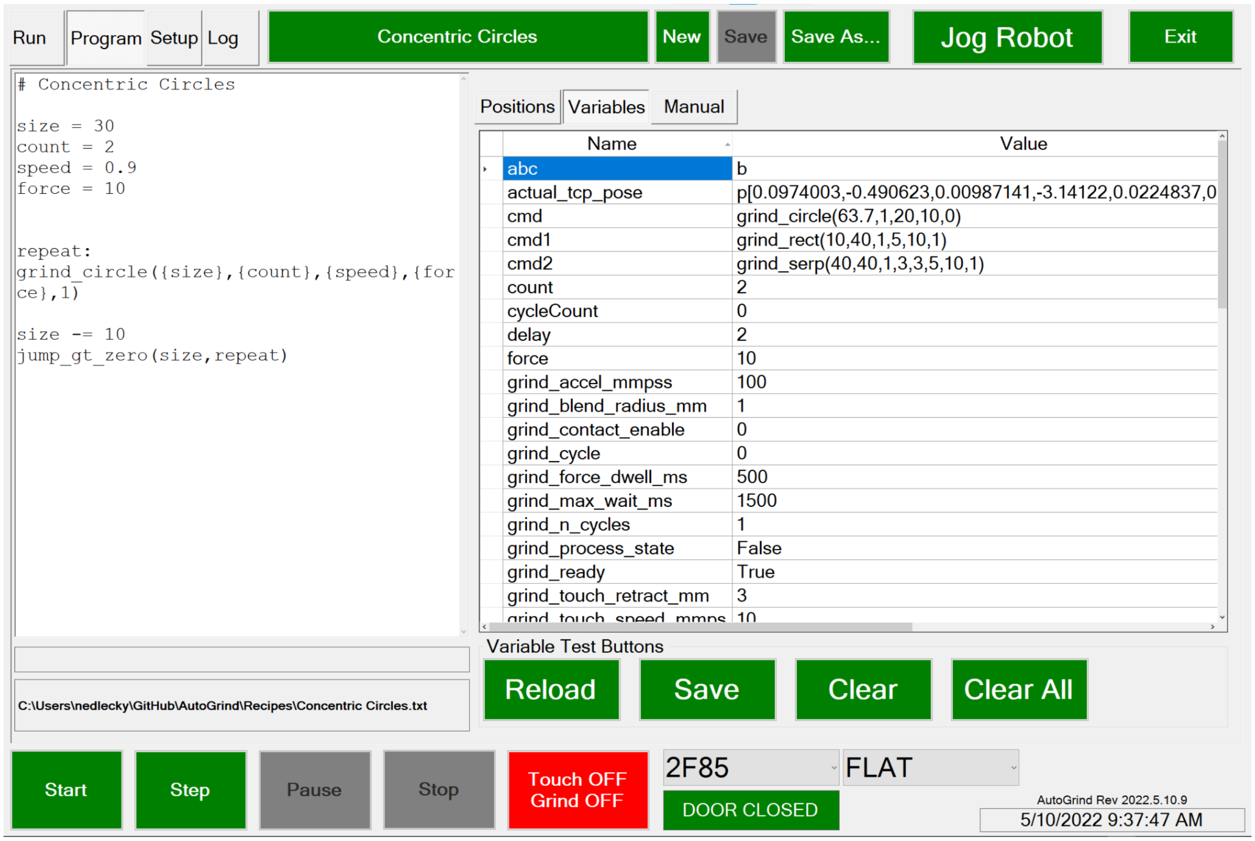
Positions can be saved manually (**Set Position**) or from the recipe with *save_position(name)*.

You can manually move to Positions in Joint (**Joint Move To Position**) or Linear (**Linear Move To Pose**) paths. These can also be executed from a recipe with *move_linear(position)* or *move_joint(position).*

Jogging is used here for setting or updating named positions or just for moving the robot. This uses the standard Jog screen.

Program Tab - Variables

Below is the **Program Tab** when the **Variables Subtab** is selected..



```
# Concentric Circles

size = 30
count = 2
speed = 0.9
force = 10


repeat:
grind_circle({size},{count},{speed},{for
ce},1)

size -= 10
jump_gt_zero(size,repeat)
```

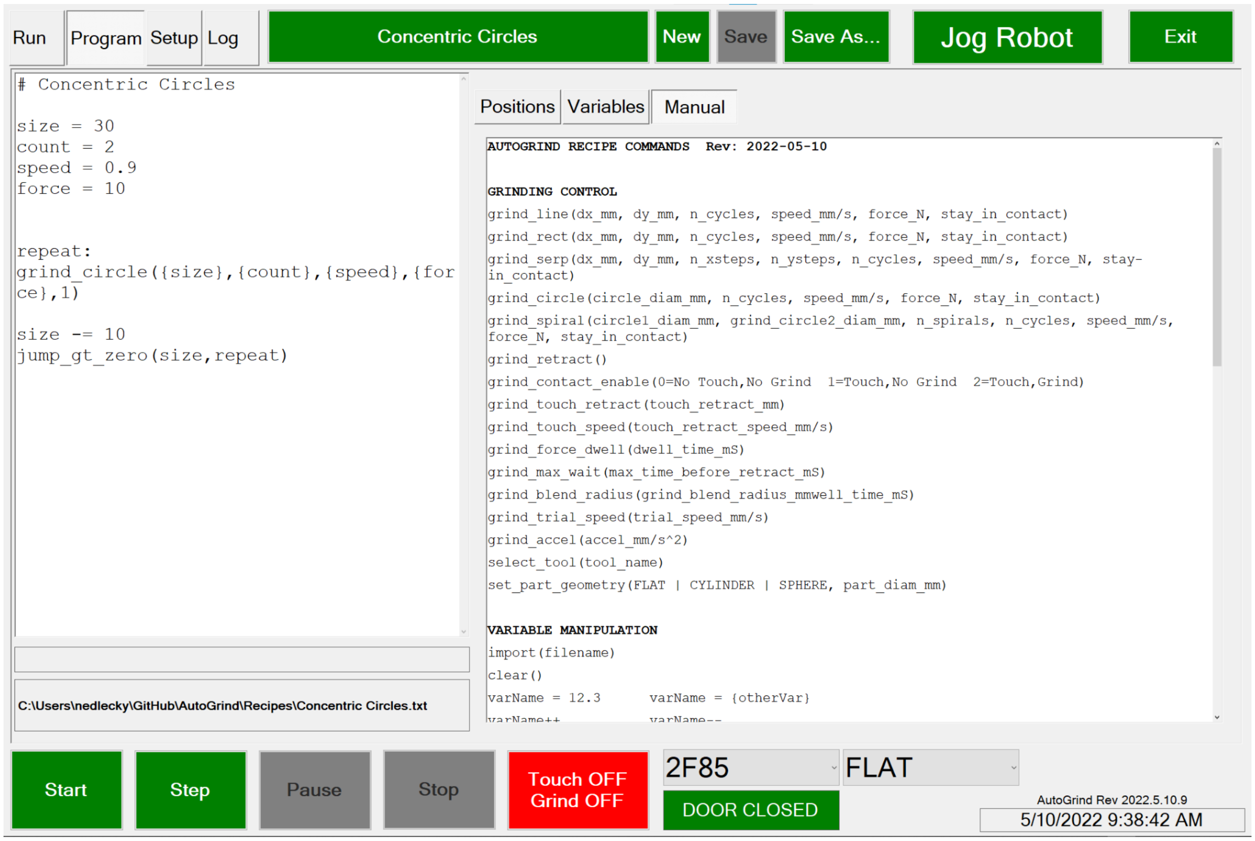
C:\Users\nedlecky\GitHub\AutoGrind\Recipes\Concentric Circles.txt

Name	Value
abc	b
actual_tcp_pose	p[0.0974003,-0.490623,0.00987141,-3.14122,0.0224837,0
cmd	grind_circle(63.7,1,20,10,0)
cmd1	grind_rect(10,40,1,5,10,1)
cmd2	grind_serp(40,40,1,3,3,5,10,1)
count	2
cycleCount	0
delay	2
force	10
grind_accel_mmpss	100
grind_blend_radius_mm	1
grind_contact_enable	0
grind_cycle	0
grind_force_dwell_ms	500
grind_max_wait_ms	1500
grind_n_cycles	1
grind_process_state	False
grind_ready	True
grind_touch_retract_mm	3
grind_touch_speed_mmps	10

This tab shows all of the local variables maintained in AutoGrind for internal, system, and user purposes. They can be edited here, too!

Program Tab - Manual

Below is the **Program Tab** when the **Manual Subtab** is selected..



```
# Concentric Circles

size = 30
count = 2
speed = 0.9
force = 10


repeat:
grind_circle({size},{count},{speed},{for
ce},1)

size -= 10
jump_gt_zero(size,repeat)
```

C:\Users\nedlecky\GitHub\AutoGrind\Recipes\Concentric Circles.txt

```
AUTOGRIND RECIPE COMMANDS  Rev: 2022-05-10


GRINDING CONTROL
grind_line(dx_mm, dy_mm, n_cycles, speed_mm/s, force_N, stay_in_contact)
grind_rect(dx_mm, dy_mm, n_cycles, speed_mm/s, force_N, stay_in_contact)
grind_serp(dx_mm, dy_mm, n_xsteps, n_ysteps, n_cycles, speed_mm/s, force_N, stay-
in_contact)
grind_circle(circle_diam_mm, n_cycles, speed_mm/s, force_N, stay_in_contact)
grind_spiral(circle1_diam_mm, grind_circle2_diam_mm, n_spirals, n_cycles, speed_mm/s,
force_N, stay_in_contact)
grind_retract()
grind_contact_enable(0=No Touch,No Grind  1=Touch,No Grind  2=Touch,Grind)
grind_touch_retract(touch_retract_mm)
grind_touch_speed(touch_retract_speed_mm/s)
grind_force_dwell(dwell_time_mS)
grind_max_wait(max_time_before_retract_mS)
grind_blend_radius(grind_blend_radius_mmwell_time_mS)
grind_trial_speed(trial_speed_mm/s)
grind_accel(accel_mm/s^2)
select_tool(tool_name)
set_part_geometry(FLAT | CYLINDER | SPHERE, part_diam_mm)

VARIABLE MANIPULATION
import(filename)
clear()
varName = 12.3      varName = {otherVar}
varName++           varName--
```

2F85

FLAT

DOOR CLOSED

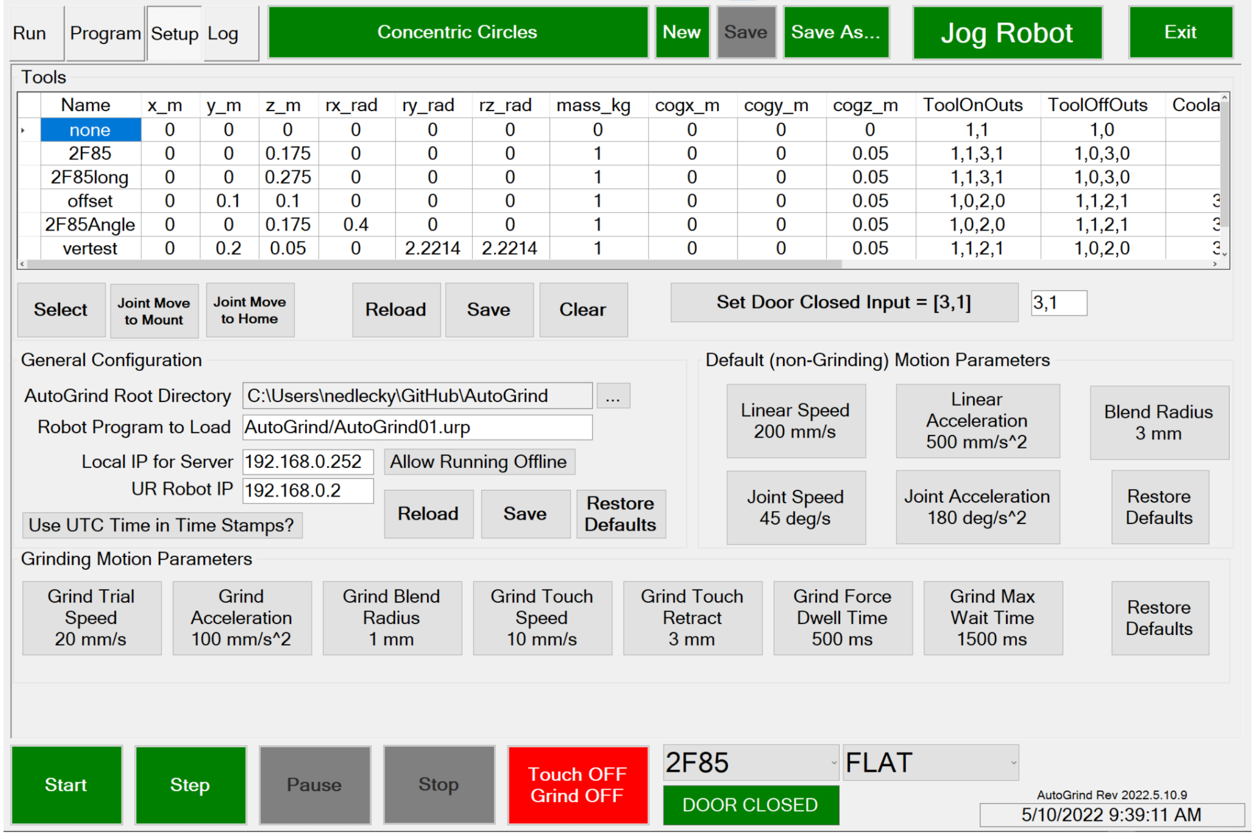
AutoGrind Rev 2022.5.10.9

5/10/2022 9:38:42 AM

This tab displays the recipe commands to help you remember what each command does and how it is called..

Setup Tab

The **Setup Tab** is where all system configuration takes place.



Tools

Name	x_m	y_m	z_m	rx_rad	ry_rad	rz_rad	mass_kg	cogx_m	cogy_m	cogz_m	ToolOnOuts	ToolOffOuts	Coola
none	0	0	0	0	0	0	0	0	0	0	1,1	1,0	
2F85	0	0	0.175	0	0	0	1	0	0	0.05	1,1,3,1	1,0,3,0	
2F85long	0	0	0.275	0	0	0	1	0	0	0.05	1,1,3,1	1,0,3,0	
offset	0	0.1	0.1	0	0	0	1	0	0	0.05	1,0,2,0	1,1,2,1	3
2F85Angle	0	0	0.175	0.4	0	0	1	0	0	0.05	1,0,2,0	1,1,2,1	3
vertest	0	0.2	0.05	0	2.2214	2.2214	1	0	0	0.05	1,1,2,1	1,0,2,0	3

3,1

General Configuration

AutoGrind Root Directory C:\Users\nedlecky\GitHub\AutoGrind ...
Robot Program to Load AutoGrind/AutoGrind01.urp
Local IP for Server 192.168.0.252 Allow Running Offline
UR Robot IP 192.168.0.2
Use UTC Time in Time Stamps?

Default (non-Grinding) Motion Parameters

Linear Speed 200 mm/s
Linear Acceleration 500 mm/s^2
Blend Radius 3 mm
Joint Speed 45 deg/s
Joint Acceleration 180 deg/s^2

Grinding Motion Parameters

Grind Trial Speed 20 mm/s
Grind Acceleration 100 mm/s^2
Grind Blend Radius 1 mm
Grind Touch Speed 10 mm/s
Grind Touch Retract 3 mm
Grind Force Dwell Time 500 ms
Grind Max Wait Time 1500 ms

2F85
FLAT

AutoGrind Rev 2022.5.10.9
5/10/2022 9:39:11 AM

Setup - General

1. **AutoGrind Root Directory:** Location where subdirectories Recipes and Logs will be created. Tools, Variables, and Positions are also stored here in the Recipes subfolder.
2. **Robot Program To Load:** Specifies the program on the UR that the UR will load and run when this software starts
3. **Local IP for Server:** This should be the IP address of the port on the host computer that is connected to the UR
4. **UR Robot IP:** The IP address that the UR is set to
5. **Allow Running Offline:** Testing only… will allow recipes to run with no UR attached
6. **Use UTC Time in Time Stamps:** Only useful for internationalization. Affect timestamps in Variables

Setup - Tools

Tools are defined in the Tool Table. Each contains the following information. These are saved in the Tools.xml file in AutoGrindRoot/Recipes and are loaded and saved automatically.

1. **Tool TCP:** This is a copy of what we would teach for the tool on the UR including x, y, z offset and rx, ry, rz orientation. Teaching these is best done on the UR and then the values simply copied to the entry in AutoGrind
2. **Mass and Center of Gravity:** Set these as you would on the UR. Accurate settings improves behavior when in freedrive mode.
3. **ToolOnOuts, ToolOffOuts:** This is a list of up to 4 digital IOs that need to be turned on or off to enable the tool. This is only done during a grind in **Touch ON Grind ON** mode. Examples: “1,1,3,1” implies that output 1 should be set to 1 and output 3 should be set to 1. “3,1” implies that output 3 should be set to 1
4. **CoolantOnOuts, CoolantOffOuts:** Similarly, these are digital output commands to be executed when grinding in **TouchOn Grind ON** mode.
5. **MountPosition:** This is a position recommended for installing/removing this tool. The system will use joint moves to approach the position with **Joint Move To Mount** or *move_tool_mount()*. This must be a position that has been defined in the **Positions Table**.
6. **HomePosition:** This is a position recommended for homefor this tool. The system will use joint moves to approach the position with **Joint Move To Home** or *move_tool_home()*. This must be a position that has been defined in the **Positions Table**.

Setup - Default Motion Parameters

Self explanatory setting for speeds and accelerations used in jogging and non-grinding motion. These are saved in the Variables.xml file in AutoGrindRoot/Recipes and are sent to the robot whenever the software starts. New values are saved automatically.

Setup - Grinding Motion Parameters

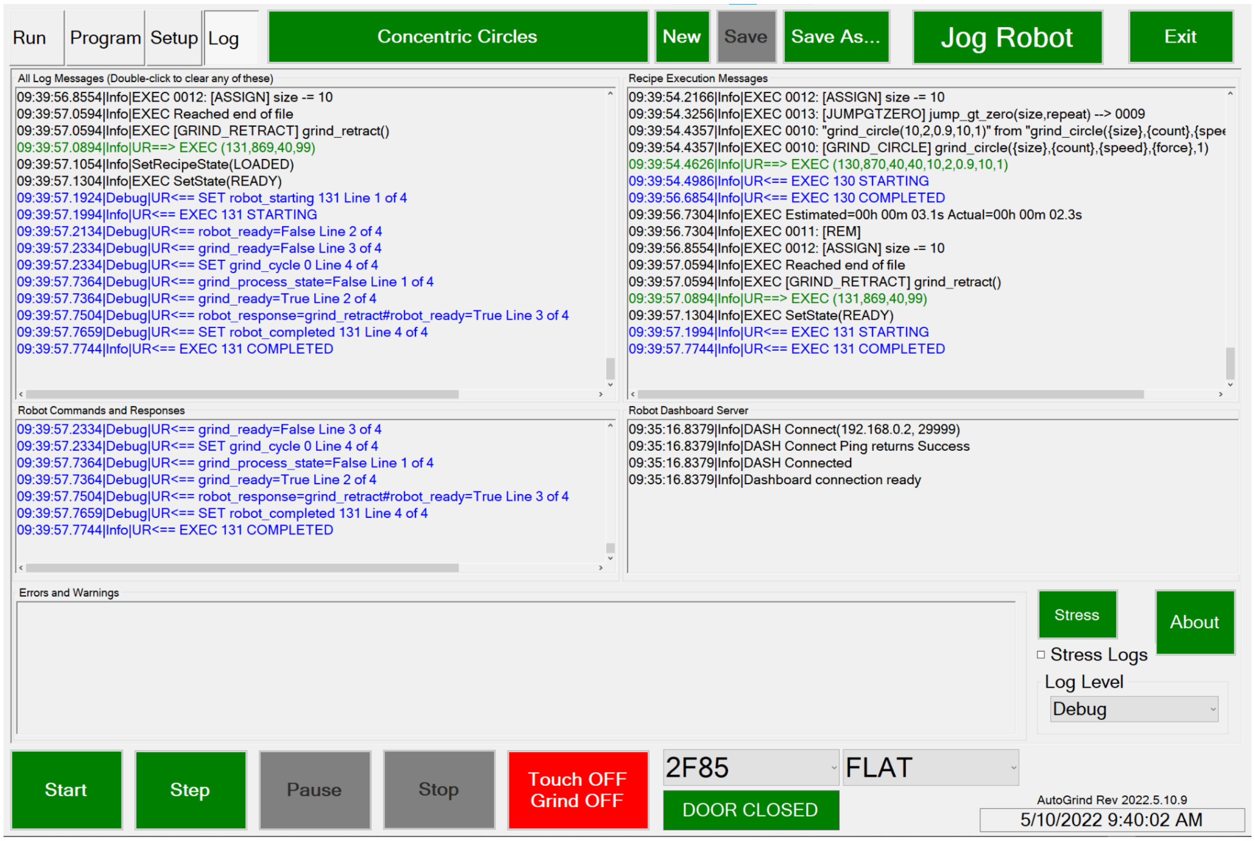
Settings governing grind operations. These are saved in the Variables.xml file in AutoGrindRoot/Recipes and are sent to the robot whenever the software starts. New values are saved automatically.
Grind Trial Speed: When not in **Touch On Grind On** mode, the grind patterns are limited to one cycle and are performed at this speed.
Grind Acceleration: Linear acceleration used during grinding
Grind Blend Radius: Blend radius used during grinding. Recommended 1mm
Grind Touch Speed: Speed robot advances toward part for touchoff. Recommended 5-10mm/s
Grind Touch Retract: Distance robot retracts from part after touchoff.
Grind Force Dwell Time: How long robot waits after turning force-on to allow time for tool to settle against part
Grind Max Wait Time: Maximum time system will wait for the next grind command if a grind command ends with 1 (stay in contact with part)

Log Tab

The Log Tab provides five windows where log messages are displayed. The level of detail in the messages is controlled by the Log Level setting:

- Error: only error messages are shown
- Warn: Error messages and Warnings are shown
- Info: All of the above, plus informational messages about execution. Default setting
- Debug: All of the above plus additional information that may be useful for debugging
- Trace: All of the above plus extremely verbose execution tracing

The All Log Messages box gets 100% of the generated messages. These messages are also written to log files in the AutoGrindRoot/Logs directory, where up to forty 25MB files are archived and deleted. Information older than this 2GB total is automatically and silently deleted.



All Log Messages (Double-click to clear any of these)

```
09:39:56.8554|Info|EXEC 0012: [ASSIGN] size -= 10
09:39:57.0594|Info|EXEC Reached end of file
09:39:57.0594|Info|EXEC [GRIND_RETRACT] grind_retract()
09:39:57.0894|Info|UR==> EXEC (131,869,40,99)
09:39:57.1054|Info|SetRecipeState(LOADED)
09:39:57.1304|Info|EXEC SetState(READY)
09:39:57.1924|Debug|UR<== SET robot_starting 131 Line 1 of 4
09:39:57.1994|Info|UR<== EXEC 131 STARTING
09:39:57.2134|Debug|UR<== robot_ready=False Line 2 of 4
09:39:57.2334|Debug|UR<== grind_ready=False Line 3 of 4
09:39:57.2334|Debug|UR<== SET grind_cycle 0 Line 4 of 4
09:39:57.7364|Debug|UR<== grind_process_state=False Line 1 of 4
09:39:57.7364|Debug|UR<== grind_ready=True Line 2 of 4
09:39:57.7504|Debug|UR<== robot_response=grind_retract#robot_ready=True Line 3 of 4
09:39:57.7659|Debug|UR<== SET robot_completed 131 Line 4 of 4
09:39:57.7744|Info|UR<== EXEC 131 COMPLETED
```

Recipe Execution Messages

```
09:39:54.2166|Info|EXEC 0012: [ASSIGN] size -= 10
09:39:54.3256|Info|EXEC 0013: [JUMPGTZERO] jump_gt_zero(size,repeat) --> 0009
09:39:54.4357|Info|EXEC 0010: "grind_circle(10,2,0.9,10,1)" from "grind_circle({size},{count},{spee
09:39:54.4357|Info|EXEC 0010: [GRIND_CIRCLE] grind_circle({size},{count},{speed},{force},1)
09:39:54.4626|Info|UR==> EXEC (130,870,40,40,10,2,0.9,10,1)
09:39:54.4986|Info|UR<== EXEC 130 STARTING
09:39:56.6854|Info|UR<== EXEC 130 COMPLETED
09:39:56.7304|Info|EXEC Estimated=00h 00m 03.1s Actual=00h 00m 02.3s
09:39:56.7304|Info|EXEC 0011: [REM]
09:39:56.8554|Info|EXEC 0012: [ASSIGN] size -= 10
09:39:57.0594|Info|EXEC Reached end of file
09:39:57.0594|Info|EXEC [GRIND_RETRACT] grind_retract()
09:39:57.0894|Info|UR==> EXEC (131,869,40,99)
09:39:57.1304|Info|EXEC SetState(READY)
09:39:57.1994|Info|UR<== EXEC 131 STARTING
09:39:57.7744|Info|UR<== EXEC 131 COMPLETED
```

Robot Commands and Responses

```
09:39:57.2334|Debug|UR<== grind_ready=False Line 3 of 4
09:39:57.2334|Debug|UR<== SET grind_cycle 0 Line 4 of 4
09:39:57.7364|Debug|UR<== grind_process_state=False Line 1 of 4
09:39:57.7364|Debug|UR<== grind_ready=True Line 2 of 4
09:39:57.7504|Debug|UR<== robot_response=grind_retract#robot_ready=True Line 3 of 4
09:39:57.7659|Debug|UR<== SET robot_completed 131 Line 4 of 4
09:39:57.7744|Info|UR<== EXEC 131 COMPLETED
```

Robot Dashboard Server

```
09:35:16.8379|Info|DASH Connect(192.168.0.2, 29999)
09:35:16.8379|Info|DASH Connect Ping returns Success
09:35:16.8379|Info|DASH Connected
09:35:16.8379|Info|Dashboard connection ready
```

Errors and Warnings

□ Stress Logs

Log Level: Debug

2F85

FLAT

AutoGrind Rev 2022.5.10.9

5/10/2022 9:40:02 AM

In addition, some messages are copied for clarity to other boxes. The boxes are labeled with their respective data: Recipe Execution Messages, Robot Commands and Responses, Robot Dashboard (Control and Monitoring) Messages, and Errors/Warnings.

Any of the boxes can be cleared by double-clicking on them. All of the messages flow into the log files and are archived as described above.

Recipe Commands

This is a copy of the recipe commands document that is available from within the software.

```
AUTOGRIND RECIPE COMMANDS  Rev: 2022-05-10

GRINDING COMMAND COMMON PARAMETERS
      dx_mm, dy_mm, diam_mm: dimensions of the patterns in mm
      n_cycles: times to repeat the pattern (ignored if test grinding)
      speed_mm/s: speed to grind at (ignored if test grinding
      force_N: force in Newtons to apply
      stay_in_contact: 0 to retract at end of grind, 1 to stay in contact

GRINDING COMMANDS
grind_line(dx_mm, dy_mm, n_cycles, speed_mm/s, force_N, stay_in_contact)
Grind in a straight line centered on the current position.

grind_rect(dx_mm, dy_mm, n_cycles, speed_mm/s, force_N, stay_in_contact)
Grind along a rectangle centered on the current position at the current RZ
angle of the tool.

grind_serp(dx_mm, dy_mm, n_xsteps, n_ysteps, n_cycles, speed_mm/s, force_N,
stay-in_contact)
Grind a serpentine pattern within a rectangle centered on the current position.
N_xsteps and n_ysteps is the number of moves needed to span the rectangle. One
or the other of these must be equal to 1.

grind_circle(circle_diam_mm, n_cycles, speed_mm/s, force_N, stay_in_contact)
Grind along a circle centered on the current position.

grind_spiral(circle1_diam_mm, grind_circle2_diam_mm, n_spirals, n_cycles,
speed_mm/s, force_N, stay_in_contact)
Grind along a variable diameter circle centered on the current position. The
circle goes from the first diameter to the second in n_spirals full
revolutions.

grind_retract()
Ensure not in contact with the part. Happens automatically if a non-grind
command is sent, if stop or pause is selected, or if grind_max_wait timer
expires.

grind_contact_enable(0=Touch OFF,Grind OFF|1=Touch ON,Grind OFF| 2=Touch
ON,Grind ON)
Set the grinding mode programmatically as shown.

select_tool(tool_name)
Setup all of the necessary environment to be able to use tool_name. No motion
is performed. Future grinds and position moves will assume this tool is
attached.

set_part_geometry(FLAT|CYLINDER|SPHERE, part_diam_mm)
Future grinds will assume the specified geometry.

The commands below provide a programmatic way to set the grinding parameters.
grind_touch_retract(touch_retract_mm)
```

```
grind_touch_speed(touch_retract_speed_mm/s)
grind_force_dwell(dwell_time_ms)
grind_max_wait(max_time_before_retract_ms)
grind_blend_radius(grind_blend_radius_mmwell_time_ms)
grind_trial_speed(trial_speed_mm/s)
grind_accel(accel_mm/s^2)
```

VARIABLE MANIPULATION

import(filename)
Open up a file and perform any variable assignment (name = value) lines found in it.

clear()
Delete all variables except ones that are marked in the Variables Table as system variables. (Variables named robot_* are automatically system variables.)

Update variables using any of these basic operations. Variables can be inserted in any command using the syntax {var_name}.

```
var_name = 12.3    var_name = {other_var_name}
var_name++         var_name--
var_name -= 17.5   var_name += 18
```

FLOW CONTROL

label:
Labels a line in a program with a name.

jump(label)
Jumps to the line after the label specified.

jump_gt_zero(var_name,label)
Jumps to the line after the label specified if the var_name is numeric and greater than 0.

end or end()
Terminate execution of a recipe.

prompt(message)
Prompt the operator with a message and pause execution until the dialog is acknowledged.

sleep(seconds)
Pause execution for the specified time. Fractional seconds may be used.

assert(var_name,value)
Testing support. Checks to see if var_name==value and generates an error message if not.

Comments
Blank lines are ignored.
Anything on a line after a '#' character is ignored.

MOTION

save_position(position_name)
The current robot position is stored in the Positions Table as position_name.

move_linear(position_name)

The robot moves along a linear path to Position position_name.

```
move_joint(position_name)
```
The robot performs a joint move to Position position_name.

```
move_tool_home()
```
Perform a joint move to the home position associated with the current tool.

```
move_tool_mount()
```
Perform a joint move to the mounting position associated with the current tool.

```
free_drive(0=OFF|1=ON)
```
Turn robot free drive mode on or off.

The commands below provide a programmatic way to set the default motion parameters.

```
set_linear_speed(speed_mm/s)
set_linear_accel(accel_mm/s^2)
set_joint_speed(speed_deg/s)
set_joint_accel(accel_deg/s^2)
set_blend_radius(blend_radius_mm)
```

ENGINEERING USE ONLY

These are all called automatically by select_tool(…), set_part_geometry(..),and the grind_...() commands. They should be used that way through the high level interface except during testing.

```
Set_part_geometry_N(1=FLAT|2=CYLINDER|3=SPHERE, diam_mm)
set_tcp(x,y,x,rx,ry,rz)
set_payload(mass_kg,cog_x_m, cog_y_m, cog_z_m)
set_door_closed_input(dig_in, value)
set_tool_on_outputs(dig_out, value,…) Can have up to 4 listed
set_tool_off_outputs(dig_out, value,…) Can have up to 4 listed
set_coolant_on_outputs(dig_out, value,…) Can have up to 4 listed
set_coolant_off_outputs(dig_out, value,…) Can have up to 4 listed
set_output(DOUT,0|1)
tool_on()
tool_off()
coolant_on()
coolant_off()
send_robot(param1,param2,…)
```

Example Recipes

Here are a few recipes that show the kinds of things that can be done in a recipe. The Testing subdirectory in the Recipes folder has many more examples that you can examine (and run!)

Remove Current Tool

Just remove the current tool from the robot. As long as the one actually mounted is selected, this goes to the tool home followed by the mount/demount position and prompts the operator when it is time to remove.

```
# Remove Current Tool
# Go through demount procedure
# Assumes you have selected whatever tool is actually mounted!

prompt(Please confirm: you wish to demount {robot_tool}?)

move_tool_home()
move_tool_mount()
prompt(Please demount tool {robot_tool})

select_tool(none)
```

Install A Tool

This goes through prompting to mount a specific tool.

```
# Install 2F85
# Example to install a tool when none is currently installed
# We just select the new tool, move to the mount position, prompt the
operator, and move to move_tool_home

# Change to whatever tool you like
tool=2F85

# Operator confirmation
prompt(About to mount {tool})

# Mounting process
select_tool({tool})  # This only informs the robot what is mounted

# This does the physical swap
move_tool_mount()
prompt(Please mount tool {tool})
move_tool_home()
```

Integrated Example

Here we start with the 2F85 tool ready to grind and swap tools and continue from the same location mid-recipe.

```
# Integrated Example
# Assumes we're where we want to grind initially but need to do a tool
swap mid-way

tool1=2F85
tool2=vertest

# Program assumes we are starting with tool1- verify internally and with
operator!
assert(robot_tool,{tool1})
prompt(Confirming tool {tool1} is currently mounted and you are grinding
on {robot_geometry})

# This will always be our grind_start position
save_position(grind_start)

# Do some grinding with tool1
move_linear(grind_start)
grind_rect(30,30,3,10,10,1)
grind_rect(20,20,3,10,10,1)

prompt(Ready to swap {tool1} to {tool2}?)
# Remove {tool1}
move_tool_home()
move_tool_mount()
prompt(Please remove {tool1})

# Install {tool2}
select_tool({tool2})
move_tool_mount()
prompt(Please install {tool2})
move_tool_home()

# Do some grinding with tool2
move_linear(grind_start) # Returns us to the starting position
grind_rect(30,30,3,10,10,1)
grind_rect(20,20,3,10,10,1)
```

Computed Concentric Circles

Here’s a test recipe that grinds 3 concentric circles explicitly and in a loop, not lifting until the final one.

```
# 26 Concentric Circle Test

# Old school
grind_circle(30,2,0.9,10,1)
grind_circle(20,2,0.9,10,1)
grind_circle(10,2,0.9,10,0)

# Do it with a loop
size = 30
count = 2
speed = 0.9
force = 10

repeat:
grind_circle({size},{count},{speed},{force},1)
size -= 10
jump_gt_zero(size,repeat)
```

Lots of Grinds

By pre-teaching points and swapping geometries, a whole day’s work could be done (other than tool swaps!)

```
# Test all the patterns on all the geometries

size1=40
size2=10
count=3
speed=5
force=10

select_tool(2F85)
cycleCount=0

redo:
move_linear(demo_flat)

set_part_geometry(FLAT,0)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(CYLINDER,400.1)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
```

```
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(CYLINDER,600.1)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(CYLINDER,800.1)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(CYLINDER,1000.1)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(SPHERE,400.2)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(SPHERE,600.2)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(SPHERE,800.2)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

set_part_geometry(SPHERE,1000.2)
grind_line({size1},{size2},{count},{speed},{force},1)
grind_line(-{size2},{size1},{count},{speed},{force},1)
grind_rect({size1},{size2},{count},{speed},{force},1)
grind_rect({size2},{size1},{count},{speed},{force},1)
grind_serp({size1},{size1},1,3,{count},{speed},{force},1)
grind_serp({size1},{size1},3,1,{count},{speed},{force},1)
grind_circle({size1},{count},{speed},{force},1)
grind_circle({size2},{count},{speed},{force},1)
grind_spiral({size1},{size2},3,{count},{speed},{force},1)

cycleCount++
jump(redo)
```