Cartographer probe installation on vanilla Klipper

Disclamer

> All actions are at your own risk. Modifying the printer in this way will invalidate the warranty! None of us are responsible for any problems associated with this manual. If you do not feel comfortable with the steps shown on this page - **STOP**.

> Before taking any step, common sense dictates that you read the **ENTIRE GUIDE** before making any changes.

Abbreviations

- *configuration file* - /config/printer.cfg file
- *macros* - /config/macros/macros.cfg file
- *homing macro* - /config/macros/sensorless_homing_override.cfg file

Legend

Line to be added.

Line to be deleted.

Introduction

From the moment I discovered Cartographer, I wanted to use it all the time. After waiting for the order and finally being at my desk, I could start the installation and configuration on my QIDI X-Max 3 with Vanilla Klipper [1], because it's always cool to have up to date software and use whatever extension/functionality you want (Spoolman! - from this point on I have a database of my stock). It was not easy, but with the help of McSneaky (he is a f*cking guru) I was able to say "It's alive! I've included all the sources at the bottom of the page if you want to dig deeper (do it - I encourage you NOT to trust me and check every point, even though the whole setup works).

To do

- set the position of the nozzle when homing to the centre of the bed (there is no effect on the bed mesh or the probe behaviour).

TL;DR

Configuration files (**blue pill**).

Use my configuration file and the macros [2] that works with the printer. Note that you'll have defined all the plugins I use in my setup (Spoolman, Crownest, etc). I encourage you to modify your printer files to match your setup. You have been warned.

Hardware

This article is not intended to guide you through the full installation process on the printer itself, but in a nutshell; download the probe holder model provided by McSneaky [3] (make sure you print in min. ABS/ASA - my suggestion - PC).

Software

(**red pill**)

Installation

Follow the official Cartographer wiki page [4]. This is straightforward and requires no change in procedure.

Configuration

This is the heart of this guide, as several things need to be modified. At first I followed the official wiki [5], but then I discovered that a few things need to be done differently.

Check your UUID (CAN version) or serial port (USB version):

USB

```
ls /dev/serial/by-id/
```

CAN

```
~/klippy-env/bin/python ~/klipper/scripts/canbus_query.py can0
```

Now we need to set the Cartographer configuration in the *configuration file*:

```
[cartographer]
serial:
#   Path to the serial port for the Cartographer device. Typically has the form
#   /dev/serial/by-id/usb-cartographer_cartographer_...
#
#   If you are using the CAN Bus version, replace serial: with canbus_uuid: and
add the UUID.
#   Example: canbus_uuid: 1283as878a9sd
#
speed: 40.
#   Z probing dive speed.
lift_speed: 5.0
#   Z probing lift speed.
backlash_comp: 0.5
#   Backlash compensation distance for removing Z backlash before measuring
#   the sensor response.
#
#   Offsets are measured from the centre of your coil, to the tip of your nozzle
#   on a level axis. It is vital that this is accurate.
#
x_offset: 0.0
#   X offset of cartographer from the nozzle.
y_offset: 21.1
#   Y offset of cartographer from the nozzle.
trigger_distance: 2.0
#   cartographer trigger distance for homing.
trigger_dive_threshold: 1.5
#   Threshold for range vs dive mode probing. Beyond `trigger_distance +
#   trigger_dive_threshold` a dive will be used.
trigger_hysteresis: 0.006
#   Hysteresis on trigger threshold for untriggering, as a percentage of the
#   trigger threshold.
cal_nozzle_z: 0.1
#   Expected nozzle offset after completing manual Z offset calibration.
cal_floor: 0.1
#   Minimum z bound on sensor response measurement.
cal_ceil: 5.0
#   Maximum z bound on sensor response measurement.
cal_speed: 1.0
#   Speed while measuring response curve.
cal_move_speed: 10.0
#   Speed while moving to position for response curve measurement.
default_model_name: default
#   Name of default cartographer model to load.
mesh_main_direction: x
#   Primary travel direction during mesh measurement.
#mesh_overscan: -1
#   Distance to use for direction changes at mesh line ends. Omit this setting
#   and a default will be calculated from line spacing and available travel.
mesh_cluster_size: 1
#   Radius of mesh grid point clusters.
mesh_runs: 2
#   Number of passes to make during mesh scan.
```

Since X-Max 3 has adxl345, I'll ignore the Input Shaper part of the configuration. The next step is to delete the `[probe]` section in the *configuration file:*

```
[probe]
pin: ^!MKS_THR:gpio21
x_offset: 28
y_offset: 4.4
#z_offset: 0.0
speed: 5
samples: 2
samples_result: average
sample_retract_dist: 3.0
samples_tolerance: 0.08
samples_tolerance_retries:3
```

Do NOT add `[safe_z_home]` as suggested in the Cartho wiki. We will use the `[homing_override]` in *homing macro*. It is more elegant.

```
```

```
[homing_override]
axes: xyz
set_position_z: 0
gcode:

    {% set home_all = 'X' not in params and 'Y' not in params and 'Z' not in
params %}              #
    {% set z_hop_speed = (printer.configfile.settings['stepper_z'].homing_speed *
30) | float %}    #
    {% set travel_speed = (printer.toolhead.max_velocity * 30) | float %}
            #
    {% set sensorless_variables = printer["gcode_macro
_Sensorless_Homing_Variables"] %}            #
    {% set z_hop_distance = sensorless_variables.z_hop_distance | float %}
               # Collect all variables needed for sensorless homing
    {% set first_homed_axis = sensorless_variables.first_homed_axis | string %}
                  # from machine config file and _Sensorless_Homing_Variables
    {% set second_homed_axis = sensorless_variables.second_homed_axis |
string %}                  #
    {% set safe_x = sensorless_variables.safe_x | float %}
   #
    {% set safe_y = sensorless_variables.safe_y | float %}
   #
    {% set safe_z = sensorless_variables.safe_z_enable | abs %}
          #

    {% if printer.configfile.settings.cartographer is defined %}
                    # Check if a third-party [cartographer] definiton is used
        {% set probe_name = printer.configfile.settings.cartographer %}
                           # If [cartographer] is found in config, set 'probe_name'
as [cartographer] config string

{% if printer.configfile.settings.beacon is defined %}
            # Check if a third-party [probe] definiton is used
    {% elif printer.configfile.settings.beacon is defined %}
 #
        {% set probe_name = printer.configfile.settings.beacon %}
        # If [beacon] is found in config, set 'probe_name' as [beacon] config string
    {% elif printer.configfile.settings.probe is defined %}
#
        {% set probe_name = printer.configfile.settings.probe %}
       # If [probe] is found in config, set 'probe_name' as [probe] config string
    {% elif printer.configfile.settings.dockable_probe is defined %}
       #
        {% set probe_name = printer.configfile.settings.dockable_probe %}
```

```
              # If [dockable_probe] is found in config, set 'probe_name' as
[dockable_probe] config string
    {% elif printer.configfile.settings.bltouch is defined %}
  #
        {% set probe_name = printer.configfile.settings.bltouch %}
         # If [bltouch] is found in config, set 'probe_name' as [bltouch] config string
    {% endif %}                                                                     #

    {% if 'probe' in printer.configfile.settings.stepper_z.endstop_pin %}
          # Check if Z is configured to home with a probe and pull config values for
        {% set probe_x_offset = probe_name.x_offset | float %}
       # X and Y offsets
        {% set probe_y_offset = probe_name.y_offset | float %}
       #
    {% else %}                                                                      #
        {% set probe_x_offset = 0 | float %}                                                #
        {% set probe_y_offset = 0 | float %}                                                # If
Z if not homed with a probe, set offsets to 0 (do not apply an offset)
    {% endif %}                                                                     #

    {% if safe_x == -128 %}                                                              #
        {% set safe_x = (printer.configfile.settings.stepper_x.position_max) /2 %}
              # If safe_x is '-128', set safe_x to the center of the X axis
    {% endif %}                                                                     #

    {% if probe_x_offset < 0 %}                                                          #
        {% set safe_x = safe_x + probe_x_offset %}
 #
    {% elif probe_x_offset > 0 %}                                                        #
Depending on if probe_x_offset is a positive or negative value, adjust safe_x
        {% set safe_x = safe_x - probe_x_offset %}
 # If the machine does not home Z with a probe, an offset is not applied.
    {% endif %}                                                                     #

    {% if safe_y == -128 %}                                                              #
        {% set safe_y = (printer.configfile.settings.stepper_y.position_max) /2 %}
              # If safe_y is '-128', set safe_y to the center of the Y axis
    {% endif %}                                                                     #

    {% if probe_y_offset < 0 %}                                                          #
        {% set safe_y = safe_y + probe_y_offset %}
 #
    {% elif probe_y_offset > 0 %}                                                        #
Depending on if probe_y_offset is a positive or negative value, adjust safe_y
        {% set safe_y = safe_y - probe_y_offset %}
 # If the machine does not home Z with a probe, an offset is not applied.
    {% endif %}                                                              #

    {% if z_hop_distance > 0 %}                                                   #
Check if z_hop_distance is greater than zero
      {% if 'x' not in printer.toolhead.homed_axes and 'y' not in
printer.toolhead.homed_axes %}    # If X and Y are not homed, move Z to
z_hop_distance
        {% if first_homed_axis != 'Z' %}
          G0 Z{z_hop_distance} F{z_hop_speed}
#
        {% endif %}
      {% endif %}                                                            #
    {% endif %}                                                              #

    {% if first_homed_axis == 'X' %}                                              # If
first_homed_axis is 'X', begin G28 param check
      {% if home_all or 'X' in params %}                                          #
        _HOME_X                                                            # If
home_all or 'X' is in params, home X
      {% endif %}                                                            #
      {% if home_all or 'Y' in params %}                                          # If
home_all or 'Y' in params, home Y
        _HOME_Y                                                            #
      {% endif %}                                                            #
    {% endif %}                                                              #

    {% if first_homed_axis == 'Y' %}                                              # If
first_homed_axis is 'Y', begin G28 param check
      {% if home_all or 'Y' in params %}                                          #
        _HOME_Y                                                            # if
home_all or 'Y' is in params, home Y
      {% endif %}                                                            #
      {% if home_all or 'X' in params %}                                          # If
home_all or 'X' in params, home X
        _HOME_X                                                            #
      {% endif %}                                                            #
    {% endif %}                                                              #

    {% if first_homed_axis == 'Z' %}
        {% if home_all or 'Z' in params %}
          _HOME_Z
        {% endif %}
        {% if second_homed_axis == 'X' %}
            {% if home_all or 'X' in params %}
              _HOME_X
            {% endif %}
            {% if home_all or 'Y' in params %}
              _HOME_Y
            {% endif %}
        {% endif %}
        {% if second_homed_axis == 'Y' %}
            {% if home_all or 'Y' in params %}
              _HOME_Y
            {% endif %}
            {% if home_all or 'X' in params %}
              _HOME_X
            {% endif %}
        {% endif %}
    {% endif %}

    {% if safe_z == True and (home_all or 'Z' in params) and first_homed_axis !=
'Z' %}             # If safe_z is true and home_all or 'Z' is in params,
        G0 X{safe_x} Y{safe_y} F{travel_speed}
 # Move to the defined safe XY location
    {% endif %}                                                              #

    {% if home_all or 'Z' in params %}                                          #
        {% if first_homed_axis != 'Z'%}
          _HOME_Z
        {% endif %}
    {% endif %}                                                              #
```

Now let's start modifying the *configuration file*.

```
[bed_mesh]
speed: 150               # Speed

speed: 500               # Speed
horizontal_move_z: 10    # Z-axis height
mesh_min: 25,10          # Minimum position of the detection point

mesh_min: 25,20          # Minimum position of the detection point
mesh_max: 315,315        # Maximum position of the detection point
probe_count: 9,9         # Number of measuring points - 6x6 - 7x7 etc.

probe_count: 15,15       # Number of measuring points - 6x6 - 7x7 etc.
algorithm: bicubic
```

```
bicubic_tension: 0.2
mesh_pps: 4, 4
```

When configuring the Z axis, make sure that you are using the virtual end stop:

```
[stepper_z]
endstop_pin: probe:z_virtual_endstop # use cartographer as virtual endstop
homing_retract_dist: 0 # cartographer needs this to be set to 0
```

We're almost done with the configuration part, you're one step away from success.

Let's add a `[PROBE_CALIBRATE]` macro in *macros* to make sure Klipper uses the correct calibration routine:

```
[gcode_macro PROBE_CALIBRATE]
gcode:
    CARTOGRAPHER_CALIBRATE
```

Bravo! Configuration is completed, we can now calibrate our brand new Cartographer probe!

Calibration

As the official calibration is not accurate in our case, we need to do some workarounds.

Home the printer using `G28 X Y`.

Move to the centre of the build plate with `G0 X162.5 Y162.5`.

At this point we cannot follow the Cartographer wiki, as our printer slides the bed all the way down (when using `G28`). There is no way to compensate for this with probe calibration. We're going to trick the printer and force it to move the bed all the way down until it touches the nozzle.

First we need to enable the `force_move` macro in *macros*.

```
[force_move]

enable_force_move: False

enable_force_move: True
```

Now we are ready to `SAVE_CONFIG` and restart the printer.

NOTE

> Now all safety limits in the Klipper configuration are **DISABLED**. Be aware that you need to be careful and do things slower rather than faster and for God's sake, observe the behaviour of your printer and make sure you have easy access to the OFF button on the back of the X-Max 3.

Going back to the calibration procedure, we are now forcing the printer to move [6] and Klipper will not be aware of the physical location of the nozzle. This is fine for now and we will correct this in a few ticks.

According to the official Klipper documentation [7], you must use the following syntax to force the printer to move:

`FORCE_MOVE STEPPER=<config_name> DISTANCE=<value> VELOCITY=<value> [ACCEL=<value>]`

My example:

`FORCE_MOVE STEPPER=stepper_z DISTANCE=-1 VELOCITY=30 [ACCEL=30]`

This will move Z up one millimetre. Slowly move to your Z0. `G28` is moving to Z10, so you need to force the Z axis to 0, then move 10 mm down with `FORCE_MOVE STEPPER=stepper_z DISTANCE=10 VELOCITY=30 [ACCEL=30]` and finally Klipper will align with the physical position of your nozzle.

We can go back to the original probe calibration process.

`CARTOGRAPHER_CALIBRATE`

This is the time to calibrate the height between the nozzle and the bed with your favourite method (the paper supplied by QIDI is fine - it works). Then `SAVE_CONFIG` and reboot the printer.

For safety reasons go to *macros*, set `enable_force_move: False` then `SAVE_CONFIG` and restart the printer.

That is all, you can start inital tests [8].

Sources

[1] https://github.com/leadustin/QIDI-up2date-english/blob/main/Klipper-Update/update+upgrade.md

[2] https://drive.google.com/file/d/12QKo8y90gV5iahb2fZBmltBkGX8aEIRE/view?usp=sharing

[3] https://www.printables.com/pl/model/692991-qidi-x3-series-beacon-cartographer-probe-low-profi

[4] https://docs.cartographer3d.com/cartographer-probe/installation-and-setup/klipper-setup

[5] https://docs.cartographer3d.com/cartographer-probe/installation-and-setup/cartographer-with-input-shaping-v2-and-v3-hybrid

[6] https://www.klipper3d.org/Config_Reference.html?h=pixel#force_move

[7] https://www.klipper3d.org/G-Codes.html#force_move_1

[8] https://docs.cartographer3d.com/cartographer-probe/installation-and-setup/cartographer-with-input-shaper-v2-and-v3-hybrid#initial-tests

---

Revision #3
Created 7 August 2024 13:10:26 by Jakub
Updated 21 August 2024 18:50:49 by Jakub