

Software interface for fiber positioner testing on LBNL fiber view camera

BigBoss-doc-174

Silber, Perry

2013-06-07

Overview

This document describes a software interface for testing fiber positioners at LBNL. The communication is between two pieces of software:

1. "master" script, coded by LBNL in Matlab or similar
2. "positioner controller" software, specific to the fiber positioner hardware, in any convenient language

Communication pipes

The communication between the two pieces of software is done via 4 ascii text files:

1. `xy_meas.txt` ... contains latest (x,y) position measured by master
2. `move_cmd.txt` ... contains latest move commanded by master
3. `motion_status.txt` ... contains flag indicating whether the positioner is in motion
4. `calibration.txt` ... contains measured calibration parameters

The positioner controller shall monitor `xy_meas.txt`, `move_cmd.txt`, and `calibration.txt` for changes, either by an event callback or by regular polling at 200-500 ms intervals. The positioner controller may not alter any of these three files; only the master may do so.

The master similarly monitors `motion_status.txt`, and may not alter it; only the positioner controller may do so.

The text files will all reside in a single convenient directory.

File specifications

For all tabular data in the files:

- Whitespace column separation
- Whitespace may be 1 or more spaces
- All numbers are decimal
- Floating point numbers are indicated by having a decimal point, followed by at least one digit
- There may be a varying number of significant digits
- Decimal points are indicated by the period symbol '.' (not by comma symbol ',')
- Integers are indicated by lack of a decimal point
- Strings are to be exactly as defined below

Line indexing and incrementation

Each new message in a file is exactly one line. At the prefix of the line are:

1. Timestamp (described below)
2. Line index number (integer)

New lines are always appended to the end of the file. The line index number increments by one with each line. Thus the latest command or message written to the file is always the last line, and always has the highest index number. Inherently, each communication file is also a complete history log.

Line indexing begins at 1. For certain tests (e.g. lifetime performance) the number of lines might exceed 65k, so code must handle line index numbers as 32 bit integer or greater.

Timestamps format

Timestamps conform to ISO 8601. They are exactly 15 characters, in the format `yyyymmddTHHMMSS`, where:

- `YYYY` ... year
- `mm` ... month
- `dd` ... day
- `HH` ... hour (24-hour clock system)
- `MM` ... minute
- `SS` ... second

EXAMPLE, the timestamp for March 6, 2013 at time 14:56:50

```
20130306T145650
```

Prefixing example

For a series of three commands named "message1", "message2", "message3", written at ~1 minute intervals, the file would look like:

```
20130306T145650 1 message1
20130306T145750 2 message2
20130306T145850 3 message3
```

File formats

The file formats are specified below. In all example code of this section, only one line is shown. The incrementing line index number will be arbitrarily indicated by the number 42. In such a case the file would in fact have 41 older entries in lines above it; line 42 would be the last and most recent in the file.

`xy_meas.txt`

- writer: master
- reader: positioner controller
- 4 columns
 - 1st column: timestamp

- 2nd column: index number
- 3rd column: floating point X value, unit mm
- 4th column: floating point Y value, unit mm
- **EXAMPLE**, where master has measured position at (3.214, -11.973) mm
 <lines 1-42 not shown>
 20130306T085055 42 3.214000 -11.97300

move_cmd.txt

- writer: master
- reader: positioner controller
- 5 columns
 - 1st column: timestamp
 - 2nd column: index number
 - 3rd column: move command string
 - Valid move command strings for all positioners are:
 - abs_xy ... move to an absolute cartesian (x,y) position, units mm
 - rel_dxdy ... move by a relative cartesian amount (Δx , Δy), units mm
 - Additionally for θ - ϕ type positioners:
 - abs_R1R2 ... rotate stages to an absolute (R1,R2) position, units deg
 - rel_dR1dR2 ... rotate stages by a relative amount ($\Delta R1$, $\Delta R2$), units deg
 - R1 is the central rotation axis
 - R2 is the eccentric rotation axis
 - Additionally for R- θ type positioners:
 - abs_TR ... move stages to an absolute (θ ,R) position, units (mm,deg)
 - rel_dTdR ... move stages by a relative amount ($\Delta\theta$, ΔR), units (mm,deg)
 - Optionally, if a positioner implements an internal homing routine, such as finding encoder limits of motor travel by hitting hardstops:
 - homing ... positioner performs its own internal homing sequence, and arguments of columns 4, 5 are disregarded
 - 4th column: floating point value for 1st coordinate, unit mm or deg
 - 5th column: floating point value for 2nd coordinate, unit mm or deg
- **EXAMPLE**, moving to the (x,y) position (-2.152,6.401) mm
 20130306T085159 42 abs_xy -2.152000 6.401000
- **EXAMPLE**, moving relatively by an amount $\Delta x = 0.013$ mm, $\Delta y = -0.002$ mm
 20130306T085209 42 rel_dxdy 0.013000 -0.002000
- **EXAMPLE**, moving to the (R1,R2) position (10° , -47°)
 20130306T085217 42 abs_R1R2 10.00000 -47.00000
- **EXAMPLE**, moving relatively by an amount $\Delta R1 = -30^\circ$, $\Delta R2 = 0^\circ$
 20130306T085230 42 rel_dR1dR2 -30.000000 0.000000
- **EXAMPLE**, tell a positioner (which is capable) to re-seek its hardstops / physical limits
 20130306T085230 42 homing 0.000000 0.000000

motion_status.txt

- writer: positioner controller
- reader: master
- 3 columns
 - 1st column: timestamp
 - 2nd column: index number
 - 3rd column: single string value indicating whether positioner is currently moving
 - moving ... positioner is currently in motion
 - stopped ... positioner is currently stopped
 - outofrange ... last move command was out of the positioner's travel range
 - belowresolutionlimit ... last move command was smaller than smallest possible step size
 - outofrange and belowresolutionlimit are a subset of, and imply, stopped
- **EXAMPLE**, where positioner is about to move, so controller sets the moving flag
20130306T085415 42 moving
- **EXAMPLE**, where positioner has completed its motion, so controller sets the stopped flag
20130306T085424 42 stopped
- **EXAMPLE**, where positioner has received a command to a target point it cannot reach
20130306T085427 42 outofrange
- **EXAMPLE**, where positioner has received a tiny relative move command to a target point closer than the smallest possible step
20130306T085430 42 belowresolutionlimit

calibration.txt

The number of columns in a calibration file depends on how many calibration values a particular positioner design requires. After the line index and timestamp, key/value pairs are added in succeeding columns.

- writer: master
- reader: positioner controller
- $2 + 2*N$ columns
 - N = number of key/value pairs
 - 1st column: timestamp
 - 2nd column: index number
- subsequent columns (3 through N)
 - i^{th} column: key ... single unbroken string token
 - $(i+1)^{\text{th}}$ column: value ... single unbroken token, type specified below
 - Keys / values for θ - ϕ positioner:
 - LENGTH_R1 / floating point number, length of R1 arm, unit mm
 - LENGTH_R2 / floating point number, length of R2 arm, unit mm
 - OFFSET_R2 / floating point number, $\phi_{\text{meas}} - \phi_{\text{nominal}}$, unit deg

- If either length is not known, its value is set to -1, and positioner controller should use its default length
 - If the ϕ offset angle is not known, its value is set to 0
 - Keys / values for other positioner designs:
 - TBD
- **EXAMPLE**, θ - ϕ positioner, neither arm has been measured yet, use defaults instead
`20130306T085307 42 LENGTH_R1 -1.00000 LENGTH_R2 -1.00000`
- **EXAMPLE**, θ - ϕ positioner, central axis arm measured to be 3.512 mm, eccentric arm 3.498 mm
`20130306T085907 42 LENGTH_R1 3.51200 LENGTH_R2 3.49800`

Typical sequence of events for a move

1. master commands fiber view camera to measure current (x,y) position
2. master updates (x,y) in `xy_meas.txt`
 (2b. positioner controller may optionally read in this measured position)
3. master updates `move_cmd.txt` with the next desired move
4. positioner controller parses the move command in `move_cmd.txt`
5. positioner controller updates `motion_status.txt` to contain the string: `moving`
6. positioner controller executes move
7. when motion is complete, positioner controller updates `motion_status.txt` to now contain the string: `stopped`
8. return to (1)

Notes on calibration

The master script will handle the position and angle offsets which occur due to mounting of the positioner hardware relative to the fiber view camera's CCD. Therefore the positioner controller may consider itself to exist in a standard Cartesian coordinate system centered at (0,0) with its x axis along 0°. The positioner controller does not need to account for position and angle offsets on the CCD.

It is assumed that the positioner implements travel limits on both axes, whether these are hard or soft limits.

As an example of typical operation of a θ - ϕ positioner, the master will:

1. request a series of `rel_dR1dR2` moves, stepping through rotations of first the central axis, and then the eccentric axis
2. calculate best-fit circles from these measurements, to infer:
 - a. (x,y) offset on the CCD
 - b. θ offset (i.e., clocking of positioner) on the CCD
 - c. lengths of R1 and R2 arms
3. write the R1 and R2 lengths to `calibration.txt`
4. begin its sequence of requesting `abs_xy` and `rel_dxdy` moves

Alternative or more elaborate calibration sequences can be handled by using different key/value pairs, as described in the [calibration.txt](#) section above. These shall be defined and added to this document based on feedback from positioner designers.