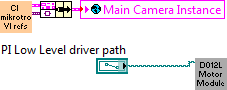
# Interfacing with BeadTracker

This document explains several use cases of connecting new labview code to the BeadTracker system. The core idea is that every module that has to run in parallel with the tracker can be added to bin/SetupConfiguration.vi, a file that is kept outside of version control and contains all functionality that is setup-specific. The SetupConfiguration file contains code that

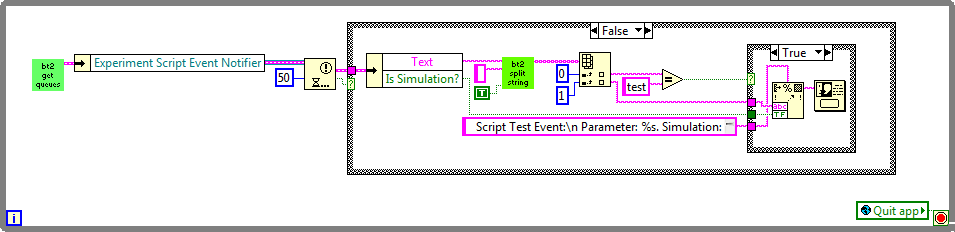
* Sets the “Main Camera Instance” global with references to the right camera Vis
* Runs the motor loop. Typically done using a first setup specific initialization (See D012L Motor Module below), which then calls the PI motor controller (TweezerTracker\bin\Modules\PIMotorController\PIMotorController.vi)
* Runs any other module that you want to start by default.

**Example SetupConfiguration.vi (Fast mikrotron camera setup)**

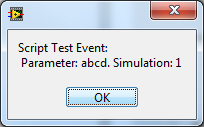
 The Mikrotron camera is linked

## Experiment Script Interfacing

The Experiment Script Notifier can be used to call external modules from within an experiment script. The following example (stored at bin/Modules/ScriptNotificationExample.vi) will be triggered when a “event test <X>” command is given.



Script “event test abcd” will result in:



## Motor Control

Motor control is done through a global queue . The scripts and user interface code puts commands on this queue, and a hardware-specific loop takes the commands off the queue and executes them.

Typically one can use  (**bin/Main/Motors/SendAxisMoveCmd.vi** ) to send motor commands for the different axes to the motor control.  will compute the constrained/clamped position as defined by offsets motor limits, and also return it as an output.

For setups with only PI motors, the queue listener loop is included in **bin/Modules/PIMotorController/PIMotorController.vi**. Motors already need to be initialized and assigned a PI system number and axis ID. See setup code for examples of this. The PI GCS functions are loaded dynamically from a directory with merged PI drivers, so they don’t cause problems with VI referring to files all over the system.



## Camera control

The BeadTracker camera system needed to be designed to support a variety of camera’s, dealing with either high framerates or very large image sizes. As a result, the camera system is fairly flexible, but also fairly complex to call. Hardware specific Camera Vis are stored as VI references in the Main Camera Instance global. This simplifies creating new camera modules, but makes it harder to update the interface unfortunately.

An example can be found in **bin/Modules/Examples/CameraGrabExample.vi**

The order is as follows:

* Turn off the camera grabbing from the main view: 
* Create the grab queues to pass to the camera Grab VI: 
* Call the grab VI reference using a ref call: 
* Run grabbing loop….
* Send stop command to camera grab VI through the camera command queue: 
* Turn the main view back on: 
* Make sure to take care of execution order as shown in the CameraGrabExample. You can’t destroy the queue before the camera grab VI is done with it.

## Tracker interfacing