USING EZCAIDL TO CONNECT TO EPICS CHANNEL ACCESS FROM SHADOWVUI FOR DYNAMIC X-RAY TRACING*

Alan Duffy#, Canadian Light Source Inc., Saskatoon, Saskatchewan, Canada

Abstract

Using the ezcaIDL library, for IDL [1], to provide an interface to EPICS [2] Channel Access through the EZCA [3] library, a simple XOP [4] extension was written that initializes ezcaIDL and thus allows access to a set of simplified IDL interface commands to connect to Channel Access from within XOP and hence from SHADOWVUI (an XOP extension) [5]. The XOP widget-based driver program is a commonly used front-end interface for computer codes of interest to the synchrotron radiation community. It models x-ray sources and characterizes optics. Extensions, such as SHADOWVUI, are optionally loaded to easily expand its functionality. SHADOWVUI is a complete Visual User Interface for SHADOW [6], which is an essential tool for x-ray optics calculations and ray-tracing. SHADOWVUI is an interactive tool for designing an optical system and visualizing results as graphs and histograms. The working scheme is to define the source and the optical elements by entering their parameters. The author has taken the usual SHADOWVUI simulation of an x-ray system a step further by using ezcaIDL to interface with the EPICS control system to access the positions of optical components in real life and then run a corresponding simulation based upon these.

INTRODUCTION

In order to predict the performance of an optical system in general and in particular a synchrotron radiation beamline, ray tracing methods are used. An essential tool for x-ray optics calculations is the ray-tracing program SHADOW, developed at Nanotech Wisconsin (University of Wisconsin), and has been used in the synchrotron community during the last 20 years. A complete Visual User Interface for SHADOW aptly named SHADOWVUI may be used as a higher level interface with graphics and menus to prepare the SHADOW inputs. It is available as an extension to another commonly used software package called XOP, a commonly used front-end interface for computer codes that model x-ray sources and optics. Essentially, the SHADOW inputs define the optical system as a collection of optical elements (mirrors, slits, screens, etc.) placed in sequential order. SHADOW generates and traces a beam from the source (e.g. bending magnet, wiggler, or indulator) sequentially through the system. The important point is that the SHADOW inputs define the optical system which usually serves to model a real synchrotron beamline. However, the parameters are static and do not change until the user enters new ones.

CONCEPT

The concept of running a dynamic x-ray tracing simulation of a beamline is straightforward (take the live positions and put them in the simulation engine), but requires some preliminary work creating the model in SHADOWVUI and determining the corresponding inputs to use from the actual beamline. This involves defining the source by supplying its parameters (e.g. energy, etc.) and defining the various optical elements with their parameters (e.g. mirror types, source plane distances, image plane distances, etc.), and how they relate to beamline parameters. The ezcaIDL library provides the tool necessary to read the beamline parameters that are maintained by the EPICS control system. The only catch is that one must define how the variables in the model are related to the parameters of the beamline. The newly developed XOP extension is used in conjunction with SHADOWVUI and requires as input a user created IDL structure defining the relationship between beamline parameters (*i.e.* process variables) and SHADOWVUI variables to make connections between the live position of the beamline optics and the variables in the simulation model.

Positioning Optical Elements

The position of each optical element in SHADOW is defined relative to the previous element (or source), not the laboratory reference frame. The user inputs the incidence and reflection angles of the central ray at each optical element as well as source and image distances to define the system. In an aligned system the central ray coincides with the optical axis, however the user has complete freedom of specifying incidence angles that are zero, positive, or larger than 90 degrees, as long as the user understands how to interpret the results. It is also not necessary for the image and source distances to correlate to the location of an actual image or object in the optical sense either. The sum of the image distance (from the previous element) and source distance simply defines the separation between optical elements in the SHADOW model. In fact, it is advantageous to think of these distances not as defining the optical element positions *per se*, but as defining the origins of their coordinate systems. Then use the mirror movement option available to place the optical components in their proper locations. Using this option to place an optical element prevents unnecessarily moving subsequent components with their positions defined relative to previous components and avoids having to recalculate distances and angles.

*Work supported by CLS
#alan.duffy@lightsource.ca

Beamline Parameters and Model Variables

The relationship between beamline parameters and the SHADOW model variables is of key importance for the dynamic simulation. A parameter in the model may be determined by one or more process variables, and vice versa. The simplest situation would be a one to one relationship between a model variable and a beamline parameter (process variable). In fact, by creating an appropriate EPICS record, one could have a single PV for each model variable used in the simulation. In any case, the user provides the relationship between the model variables and process variables as strings defining the equation(s) that relates them as part of a PV_INFO structure as illustrated in Fig. 1.

```
«struct»
ENERGYFEEDBACK : PV_INFO
pv = 'BL1606-B1-1:Energy:fbk'
desc = 'Beamline Energy [eV]'
pv_min = 1500
pv_max = 10000
oe_num = 0
src_num = 0
pv_2vui =
'(*ptrSRC).PH1 = beamline.EnergyFeedback.vui_val
& (*ptrSRC).PH2 = beamline.EnergyFeedback.vui_val'
vui_2pv = '0.5*((*ptrSRC).PH1+(*ptrSRC).PH2)'
vui_val = 0.0
```

Figure 1: Example PV_INFO structure.

All the information needed for the simulation is stored in an IDL structure named beamline with a field for each process variable containing a nested PV_INFO structure. The fields of this nested structure are described in table 1.

Table 1: PV_INFO structure content

Field	**Type**	**Description**
pv	string	EPICS process variable string
desc	string	Text to describe process variable
pv_min	float	Lower limit
pv_max	float	Upper limit
oe_num	int	Optical element number (zero otherwise)
src_num	int	Screen number (zero otherwise)
pv_2vui	string	Equation(s) to convert value of PV(s) to SHADOWVUI variable
vui_2pv	string	To convert value of SHADOWVUI variables(s) to PV value
vui_val	float	Stores SHADOWVUI variable value

In the example show in Fig. 1, the beamline parameter is the energy feedback. The process variable string 'BL1606-B1-1:Energy:fbk,' is stored and retrieved from the IDL variable *beamline.ENERGYFEEDBACK.pv*, and so on. A SHADOWVUI workspace stores its parameters in a structure in a state variable that includes variables one finds in the start.xx files. The start.00 file, for example may have the line: *PH1 = 5000.00000.* The corresponding SHADOWVUI variable is the rather obtuse looking (*(state.ptrsrc)[state.ifc.src_sel]).PH1. In order to simplify things, the ezcaShadowVUI extension defines ptrSRC, ptrOE1, ptrOE2… as pointers to the source parameters, and optical element parameters. As these are pointers, the dereference operator '*' must be used as appropriate. For example the bend radius of optical element 1 is *(*ptrOE1).RMIRR*.

The entire structure with all the information for the process variables that are to be incorporated into the simulation must be defined. This may be done by creating the structure in an IDL session and then saving it to an IDL file such as *pv_defs.sav*, or by creating a file with the commands to define the structure and executing it from SHADOWVUI using the *xop_macro_compact* command. The former method requires the user to restore the .sav file, with the command: *restore, 'pv_defs.sav'*. Once the variable is created or restored in a SHADOWVUI macro, the user can then call: *reshadowvui, beamline.* This will start up a widget similar to the one in Fig. 2.

EZCASHADOWVUI WIDGET

The ezcaShadowVUI widget has a tab for the source, and each optical element, and sub-tabs for each screen. The widget shows live process variable values and the equivalent SHADOWVUI value calculated from the vui_2pv string field.

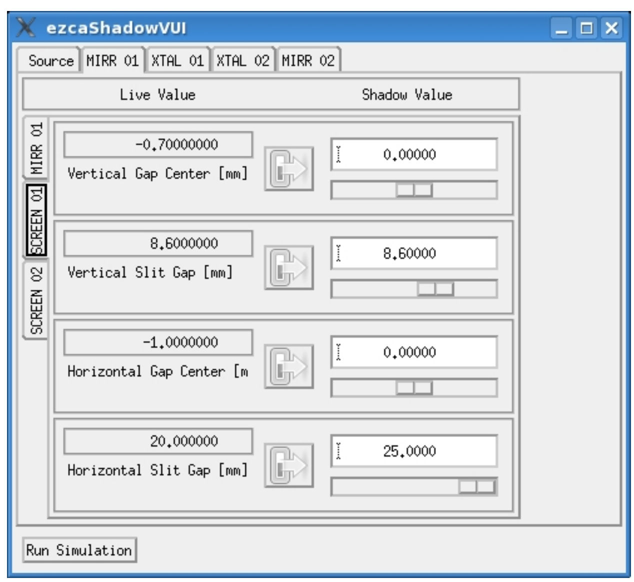

Figure 2: ezcaShadowVUI widget interface.

The ezcaShadowVUI extension uses the ezcaIDL channel access features to set up process variable monitors so that the live values update once per second. The equivalent SHADOWVUI value is shown in an editable box with a slider. The user can click a button to copy the live value to SHADOWVUI, doing so executes the command stored in the pv_2vui string field. After the user has finished copying live values and/or editing the SHADOWVUI values, the *Run Simulation* button may be pressed to execute the source generation and ray trace routines and to show a plot of the beam focus. Also the ezcaShadowVUI widget is non-modal, meaning that the user can do other things with the main SHADOWVUI interface while this widget is running. In particular, the user still has access to all the usual features of SHADOWVUI, such as running macros. The widget code can be modified to allow automatically running a macro to be triggered either by the *Run Simulation* button or a process variable event.

Dynamic Ray Tracing

The copying of live values to the simulation engine, as well as running the source generation and a ray trace with a new plot, may be set up to be down automatically for true dynamic ray tracing. There is a limitation, however, on the rate at which the ray tracing can occur. It is possible to decrease the number of rays to increase the simulation speed, but this may not be desirable. However, a refresh rate of a new trace every few seconds should not pose a problem.

Installation

The installation of the XOP extension ezcaShadowVUI is done by creating an ezcashadowvui folder in the XOP extensions directory and copying the ezcashadowvui.sav file into it. EPICS should be installed with the extensions [3] ezca, ezcaIDL, and EzcaScan so that the appropriate libraries are available. Finally, the EZCA_IDL_SHARE environment variable should point to the location of the libezcaIDL.so file. The ezcaShadowVUI extension will be made available in the near future.

CONCLUSION

The ezcaShadowVUI extension is in its preliminary stage of development, but has proven useful in modelling a real beamline, the SXRMB beamline at the Canadian Light Source, and should prove useful in modelling other beamlines. The most time consuming part of setting up the dynamic simulation is determining the relationship between SHADOW variables and beamline parameters and then creating the structure to contain that information. However, it is just a matter of reconciling the different coordinates systems of one with the other. The real challenge as always is to properly understand the beamline and in particular the intricate details of pivot points and rotation axis. In order to have a truly accurate simulation, it is important to be attentive to these details.

REFERENCES

[1] Interactive Data Language (IDL) by ITT Visual Information Solutions, 4990 Pearl East Circle, Boulder Colorado 80301, United States of America; http://www.ittvis.com/.

[2] http://www.aps.anl.gov/epics/.

[3] http://www.aps.anl.gov/epics/download/extensions/

[4] Roger J. Dejus and Manuel Sanchez del Rio, “XOP: a multiplatform graphical user interface for synchrotron radiation spectral and optics calculations,” Proc. SPIE, Vol. 3152, 148 (1997).

[5] http://www.esrf.eu/UsersAndScience/Experiments/TBS/SciSoft/xop2.3/extensions/

[6] C. Welnak, G.J. Chena, and F. Cerrinaa, “SHADOW: A synchrotron radiation and X-ray optics simulation tool,” Nucl. Instr. and Meth. in Phys. Res. A, Volume 347, Issues 1-3, 11 August 1994, Pages 344-347