Abstract:
The PEXLib5.2 draft specification distributed by X consortium is under public review. In the specification, new function interfaces are defined. New primitives, such as cone and sphere are provided. Also some new rendering attributes, such as transparency and texture mapping, are supplied.

As my thesis project, I implemented the PEXlib5.2 OCC style functions defined in the PEXlib5.2 draft specification based on PEXlib5.1. With the emulation functions, user's application can run in the PEX5.1 environment just as in the PEX5.2 environment. In this paper, I introduce some background of PEX and PEXlib, analyze the features of PEXlib5.2, and then describe how to implement the PEXlib5.2 functions based on PEXlib5.1

PEXlib5.2 EMULATION IMPLEMENTATION

by

Kai Wang

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

November, 1995

N378
W1856

APPROVAL

of a thesis submitted by

Kai Wang

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

1/2/96	[illegible]
Date	Chairperson, Graduate Committee

Approved for the Major Department

1/2/96	[illegible]
Date	Head, Major Department

Approved for the College of Graduate Studies

1/7/96	[illegible]
Date	Graduate Dean

STATEMENT OF PERMISSION TO USE

Signature [illegible]
Date 12/29/95

TABLE OF CONTENTS

List of Tables

List of Figures

ABSTRACT

The PEXlib5.2 draft specification distributed by X consortium is under public review. In the specification, new function interfaces are defined. New primitives, such as cone and sphere are provided. Also some new rendering attributes, such as transparency and texture mapping, are supplied.

As my thesis project, I implemented the PEXlib5.2 OCC style functions defined in the PEXlib5.2 draft specification based on PEXlib5.1. With the emulation functions, user's application can run in the PEX5.1 environment just as in the PEX5.2 environment. In this paper, I introduce some background of PEX and PEXlib, analyze the features of PEXlib5.2, and then describe how to implement the PEXlib5.2 functions based on PEXlib5.1.

INTRODUCTION

X Window System And PEX

X Window System is a network-oriented windowing system. It provides a network-transparent and vendor-independent operating environment for distributed graphics applications.

X Window System uses a client-server architecture. X server is the piece of the program that controls the display resources, such as keyboard, mouse, and one or more display screens. X client is the user's application program. A client connects with X server to request services or responds to events through X network protocol. The network protocol interface is designed to work either within a single CPU or between CPUs. So the user's client can run locally or on a remote host that can connect with the server. To mask the detail of X network protocol, a C language function package known as Xlib is provided with the X Window System. Xlib is the application program interface to the X network protocol: Clients communicate with the server using functions in Xlib. The X Window System architecture is shown in Figure 1.

The X Window System only provides text and two_dimensional graphics. PEX is an extension to the X Window System for supporting three_dimensional graphics. PEX provides all the common features found in most modern 3D graphics system, but provides them in a way that is seamlessly integrated with X. In fact, PEX is an extension to the X network protocol. As in the X protocol, a PEX application client communicates with the PEX server using the PEX protocol. The PEX server contains the PEX server extension, which receives and interprets the PEX protocol messages and executes the PEX requests. Just as Xlib is the application program interface to the X protocol, PEXlib subroutine package is provided as program interface to PEX protocol. The applications use functions in PEXlib to create and send PEX protocol messages. Figure 2 shows the relationship between X and PEX.

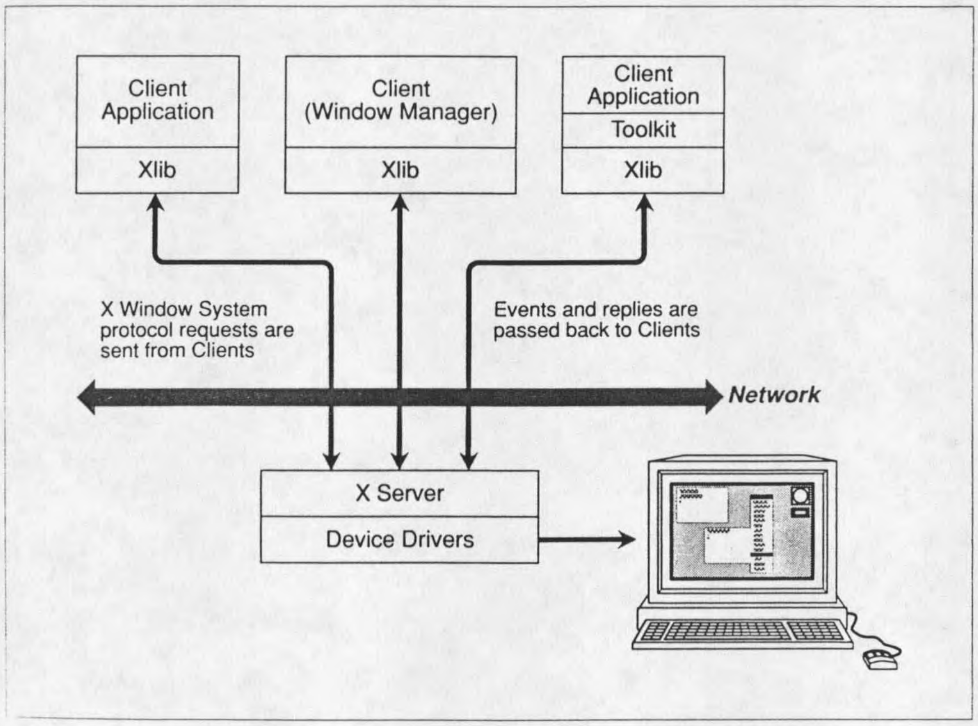

Figure 1: The Architecture of X Window System

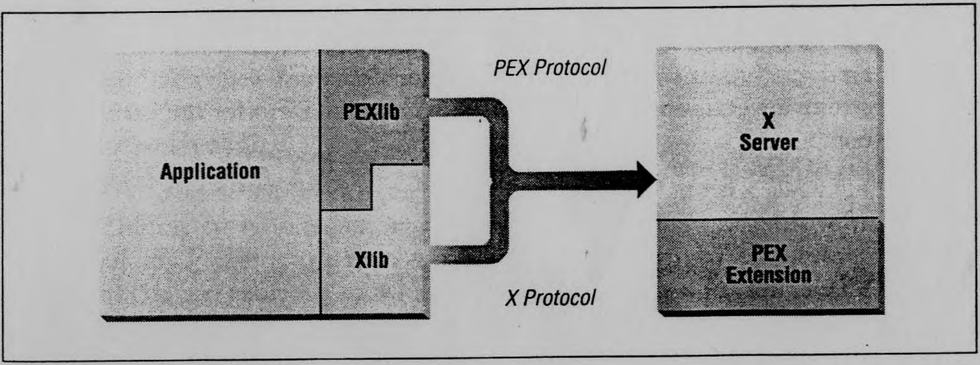

Figure 2: Relationship Between X And PEX

PEXlib 5.1

PEXlib5.1 is a C programming interface for PEX protocol. It is a high-level graphics library. In PEXlib a graphic image is composed of primitives and their attributes. Primitives are basic graphic objects like lines and polygons. Attributes control the appearance of primitives, such as color and orientation. An application sends output commands by using PEXlib functions to have PEX server render the primitives and their attributes. Figure 3 shows a high level view of PEX.

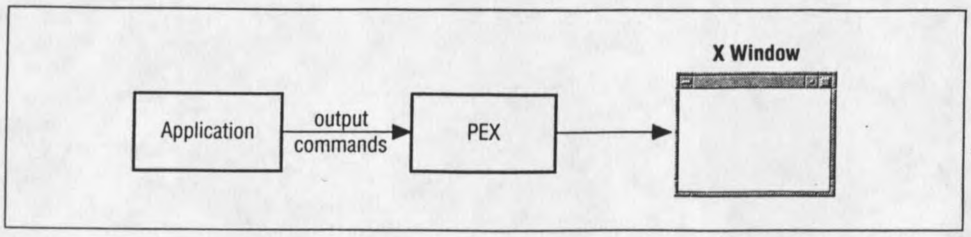

Figure 3: High Level View of PEX

There are two forms for rendering in PEX: immediate mode and structure mode. In immediate mode, the server renders primitives as soon as it receives them. The application must send the primitives and attributes that comprise the image each time the picture is generated. In structure mode, primitives, attributes and other information can be stored for later processing. The image can be regenerated simply by redrawing the contents of each structure. Figure 4 shows a renderer and a structure store inside PEX.

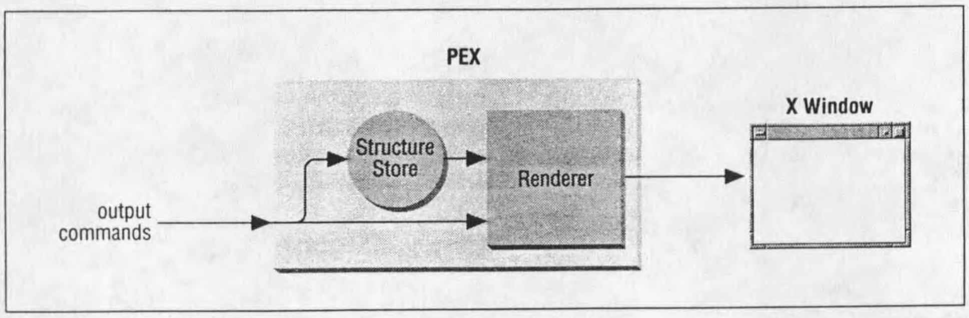

Figure 4: Renderer And Structure Store Inside PEX

PEXlib 5.1 is PEXlib fourth revision. It contains over 200 functions. Table 1 list the categories of PEXlib 5.1 functions.

Table 1: PEXlib5.1 Function Categories

Category	Description
Startup and features	Initialize PEX, determine the capabilities of PEX servers.
Renderers	Create renderer, operate on renderer's attributes, draw primitives without storing them internally.
Structures	Create structure, operate on structures and strore primitives in structure.
Line primitives and attributes	Generate polyline and polyline set output primitives and set their attributes.
Area primitives and attributes	Generate polygons and other multifaceted output primitives and set their attributes.
B-splines and attributes	B-spline curve and surface primitives and their attributes.
Texts and attributes	Draw texts and set their attributes.
Markers and attributes	Draw Marks and set their attributes.

Table 1:PEXlib5.1 Function Categories(continue)

Category	Descripttion
Color	Define colormaps, set color approximation, and other color-related functions.
Lighting and shading	define light sources, the reflective properties of primitives, and specify shading smoothness.
Modeling and transformations	Use geometric transformations and hierarchy to model objects and collections of objects.
Viewing	Look at models from different perspective.
Lookup table	operations on PEX lookup table resource.
Pipeline context	Functions on PEX pipeline context resource.
Name sets	Functions on PEX name set resource.
Picking	select primitives from those displayed on the screen.

Table 1:PEXlib5.1 Function Categories(continue)

Category	Descripttion
Spatial searches	Search a structure network for primitives that lie within a given region of world coordinates.
Font	Select different fonts for text.
Miscellaneous utilities	Other utility functions provided by PEX.

PEXlib5.2 AND ITS FEATURES

X consortium distributed a draft specification of PEXlib5.2 for public review. There are many new features of PEXlib 5.2 provided in the draft specification. A more convenient programming interface called Output Command Context is used for output primitives. Graphic data can be supplied in flexible data formats. In the specification, new primitives are added, such as arc, sphere and cone. New rendering attributes are provided, such as texture mapping, alpha blending and transparency.

Output Command Context(OCC) Interface

In PEXlib 5.1, the interface for the output command functions is defined as the explicit interface. The explicit interface uses the same first three arguments: display, which specifies the display connection; resource_id, which specifies the resource identifier for the targeted renderer or structure; and req_type, which specifies whether the application renders the output commands immediately, or stores the output commands in a structure. The explicit interface requires that you specify all arguments on every output command function you invoke.

In the PEXlib5.2 specification, a new interface for the output command functions, Output Command Context, is defined. In the Output Command Context interface, a data structure called Output Command Context (OC Context) is used to replace the first three arguments in PEXlib5.1 output command functions. The OC Context is an opaque structure that contains many of the arguments which are commonly found in output command functions. The user's application can use a set of special OC manipulation functions to modify the fields in the opaque OC Context, and then use the OC context in subsequent invocations of output command functions.

The Output Command Context interface has far fewer arguments than the PEXlib5.1 explicit interface, making coding easier and improving performance. In order to guarantee backwards compatibility, the explicit interface defined for PEXlib5.1 is included in the PEXlib5.2 specification. Output command functions defined in PEXlib5.2 have two forms: the original PEXlib5.1 format and OC Context format. But all the new primitive functions defined in the PEXlib5.2 specification only have the OCC form.

Flexible Data Format

In the PEXlib5.2 specification, data format flexibility is provided for primitives that have facet and/or vertex data parameters. We can supply the graphical data (coordinates, vertex attributes, facet attributes, and floating-point data) in three different forms. The three different forms are packed, stride, and unpacked.

- **Packed Data Form**

  The packed form requires user to format the data into packed data structures defined in PEXlib. This is the only form supported in PEXlib5.1. When passing the facet or vertex data structure to OCC style functions, we need to set the surface_attributes, surface_vertex_attributes, line_vertex_attributes, color_type, and data_model fields in OC Context with correct values.

  For PEXlib5.2, sometimes we need to use facet or vertex data that include floating point data. But PEXlib does not provide a set of structure type definition for the data form. Hence, we need to design our own data structure to pass the facet or vertex parameters. There are required orders for the structure definition:

For facet data, the required order is:

PEXColor *	one of the PEXlib color types, if provided.
PEXVector	normal, if provided.
float[n]	floating point data, if provided.

For vertex data, the required order is:

PEXCoord	Center.
PEXColor *	one of the PEXlib color types, if provided.
PEXVector	normal, if provided.
unsigned int	edges, if provided.
float[n]	floating point data.

- **Stride Data Form**

  The stride data form is more flexible. It allows the user to supply facet or vertex data formatted in application-defined structured arrays without the need to copy the data into the PEXlib-defined structures when invoking the PEXlib functions. Each structure in the structured array corresponds to data for a single facet or vertex along with other application specific data.

  When we want to use the stride data interface, we set data_model in OC context to PEXDataStride, and set the OC context values we have to set for packed. In addition, we need to set facet_stride or vertex_stride in stride structure to the size of each array element. and set the offset members to the offsets of the corresponding facet or vertex data items within the array element. Figure 5 shows the vertex stride data model.

- **Unpacked Data Form**

  The unpacked data interface allows user to supply the facet or vertex data in separate lists for each data type. When using this data interface,

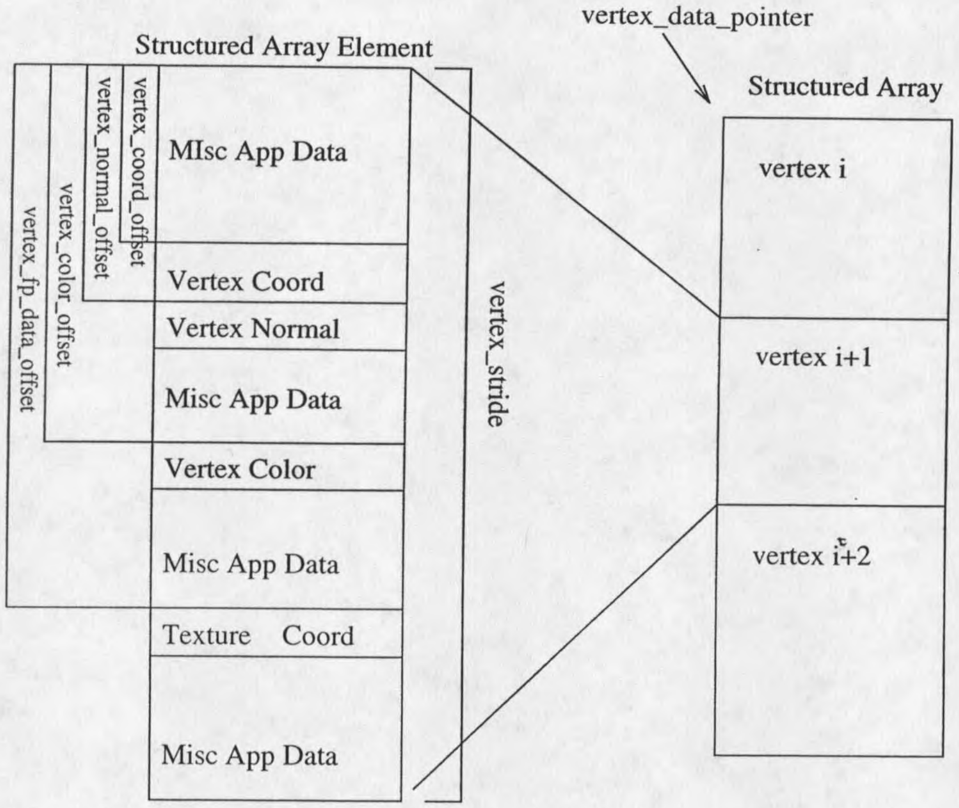

Figure 5: Stride data model.

we set the data_model OCC member to PEXDataUnpacked and set the corresponding size fields in the OCC. In addition, we need to place the pointers to each of the data arrays in PEXUnpackedFacetData or PEX-UnpackedVertexData structure and pass the address of the structure. Figure 6 shows the vertex unpacked data model.

Primitives

In PEXlib5.2, primitive functions defined in PEXlib5.1 have two forms: the original PEXlib5.1 format and the corresponding OC Context style function. Some primitives simply have their OCC version of the functions. Some primitives are reorganized into new OCC functions so that one OCC function can serves the role of several non-OCC function. Table 2 lists the relationship between OCC and non-OCC primitive functions.