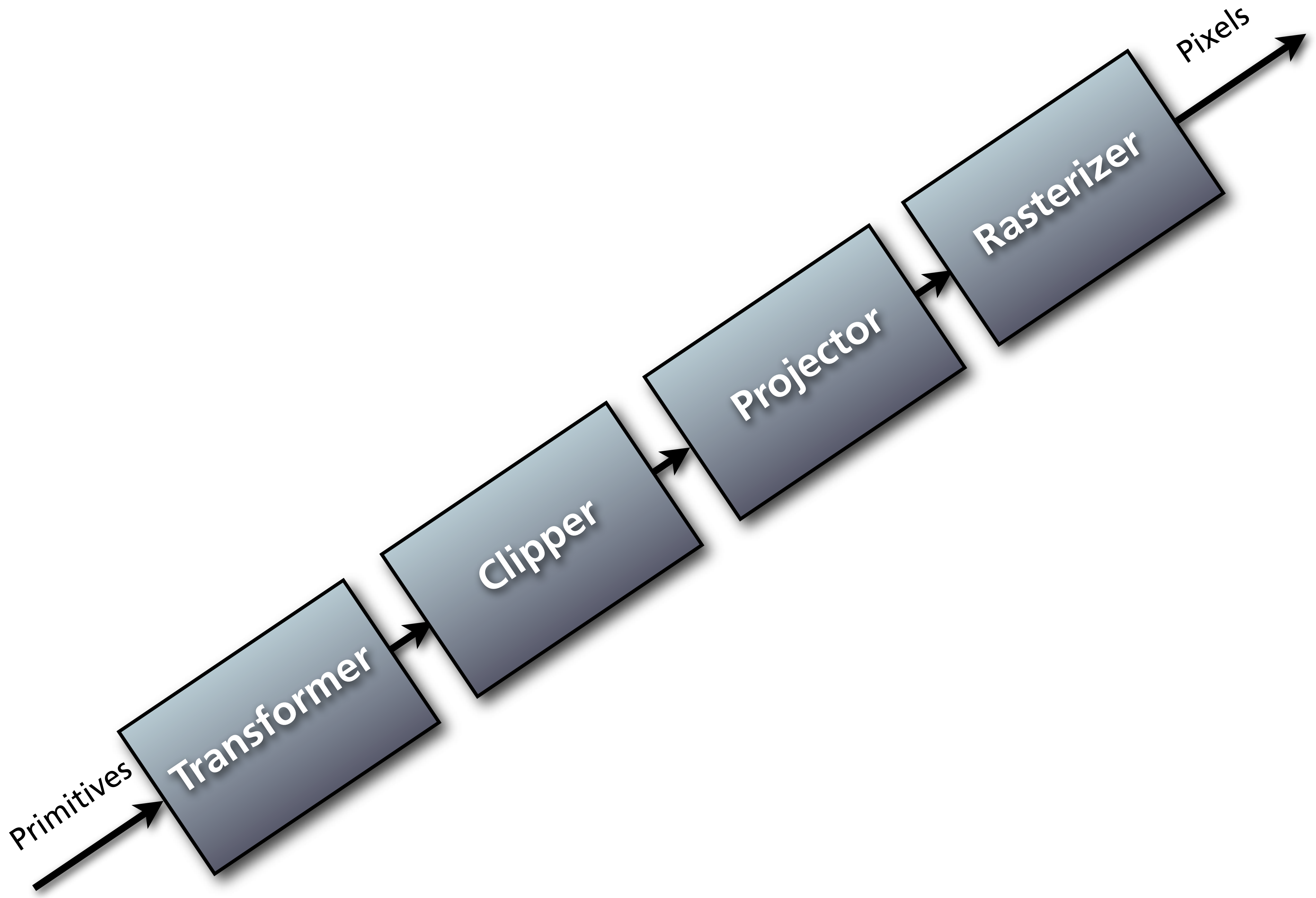


Monash University • Clayton's School of Information Technology

CSE3313 Computer Graphics

Lecture 7: OpenGL — Windows, Display Lists, Pipline

Scaling to the Window

Reference Angel, Ch. 2

- Whenever the window size changes, the viewport and clipping volume must be redefined to preserve the same aspect ratio.
- The GLUT provides the function `glutReshapeFunc`, which tells the program which function to call when the window dimensions change (This works the same way as `glutDisplayFunc`).
- Suppose our function is called `ChangeSize`. The function would have the following prototype

  ```
  void ChangeSize( GLsizei w, GLsizei h )
  ```

  where the windowing system would pass the width of the resize window via the parameter `w` and the height `h`.
- Prior to the call on `glutMainLoop()` in the main routine there would be a call of the form

  ```
  glutReshapeFunc( ChangeSize )
  ```

ChangeSize Example

-
  ```
  void ChangeSize( GLsizei w, GLsizei h ) {
      /* prevent a divide by zero */
      if ( h == 0 ) h = 1;

      /* make the viewport as big as the window */
      glViewport( 0, 0, w, h );

      /* reset the coordinate system */
      glMatrixMode( GL_PROJECTION );
      glLoadIdentity( );

      /* Establish the clipping volume
         left, right, bottom, top, near, far */
      if (w <= h )
         gluOrtho2D( 0.0f, 250.0f, 0.0f, 250.0f*h/w );
      else
         gluOrtho2D( 0.0f, 250.0f*w/h, 0.0f, 250.0f );
      glMatrixMode( GL_MODELVIEW );
      glLoadIdentity();
  }
  ```

Assumes the view volume originally has a 250 x 250 square cross section

Display Lists

- In **immediate mode** as soon as a primitive is output, it is displayed and the system retains no memory of it.
- A second mode is **retained mode** graphics. The object is defined once and the definition is placed into a **display list**. A single function call can be used to cause the object to be redisplayed.
- Each display list must have a unique identifier.

```
#define BOX 1 /* unused list number */
glNewList( BOX, GL_COMPILE );
  glBegin( GL_POLYGON );
    glColor3f( 1.0, 0.0, 0.0 );
    glVertex2f( -1.0, -1.0 );  glVertex2f( 1.0,
-1.0 );
    glVertex2f( 1.0, 1.0 );  glVertex2f( -1.0, 1.0 );
  glEnd();
glEndList();
```

Display Lists (cont.)

- `GL_COMPILE` tells the system to send the list to the server but not to display its contents.
- To draw the box, we call the function:

  ```
  glCallList( BOX );
  ```

- The output primitives get drawn according to the current values of attributes like current colour, current transformation matrix.
- A standard procedure is to always push attributes and matrices on to their own stack when the display list is entered and to pop them off the stacks when the display list is exited.
  At the beginning of the display list:

  ```
  glPushAttrib( GL_ALL_ATTRIB_BITS );
  glPushMatrix( );
  ```

Display Lists (cont.)

- At the end of the display list:

```
    glPopAttrib( );
    glPopMatrix( );
```

- Other GL display list calls

```
    glGenLists( number );
```

  returns the first integer of `number` consecutive integers that are unused labels.

```
    glCallLists( ... );
```

  executes multiple display lists with a single function call. glCallList can appear inside a display list. To avoid the possibility of infinite recursion, a limit is placed on the nesting level of display lists during display list execution. This limit is at least 64 and depends on the implementation.

- A display list can be executed between a call to `glBegin` and `glEnd` as long as the display list only includes commands that are allowed between `glBegin` and `glEnd`.

Keyboard Interaction

- Pressing a key on the keyboard queues a keyboard event. The callback function `myKeyboard()` is registered with that type of event through

  ```
      glutKeyboardFunc( myKeyboard );
  ```

  The function must have the prototype:

  ```
  void myKeyboard( unsigned char key, int x, int y );
  ```

  `key` is used to pass the ASCII value of the key pressed.

  The values of `x` and `y` report the position of the mouse at the time the event occurred. `y` is actually the number of pixels down from the top of the screen.

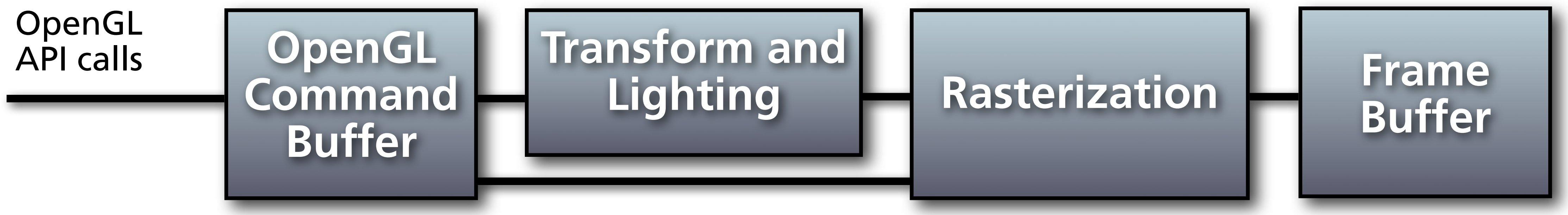
- Implementations of `myKeyboard` often consist of a large `switch` statement with a `case` for each key of interest.

- Names for special keyboard keys, such as function keys, arrow keys can be found in `glut.h`

  e.g: `GLUT_KEY_F3, GLUT_KEY_LEFT, GLUT_KEY_PAGE_UP`

  (For these keys use `glutSpecialFunction()`)

The OpenGL Pipeline



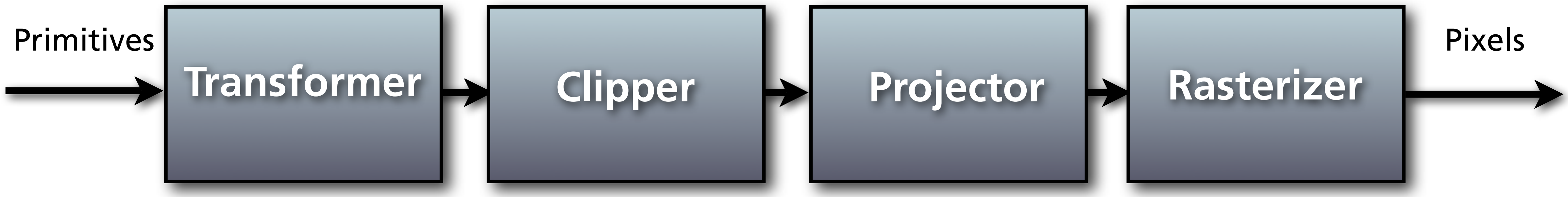
- As an application makes OpenGL calls, the commands are placed in a **command buffer**. When the buffer is flushed, the commands and data are passed to the next stage in the *pipleine*.

- Vertex data is **transformed** and **lit**. This may require lots of calculation. Lighting calculations are carried out to determine how bright the colours should be at each vertex.

- In the **Rasterization** stage the colour image is created from geometric, colour and texture data.

- The image is then placed in a **frame buffer** — the memory of the graphics device.

The OpenGL Pipeline



- Early hardware only did rasterization. Now transforms, lighting and shading are done in hardware. Some hardware supports programable shading.
- OpenGL is an immediate mode API. Each command has an immediate effect on the current rendering state. Each state is either on or off or contains some numeric value.
- OpenGL can be seen as defining a **state machine.**
- Further Information:

OpenGL programming guide : the official guide to learning OpenGL, version 1.4 / OpenGL Architecture Review Board, Mason Woo [et al.] Reading, MA : Addison-Wesley, 2004.

OpenGL reference manual : the official reference document for OpenGL, version 1.4 / OpenGL Architecture Review Board. Reading, Mass. : Addison-Wesley, 2004