

Teaching Graphics Using Ada

Dr. Wayne Brown

Assistant Professor of Computer Science

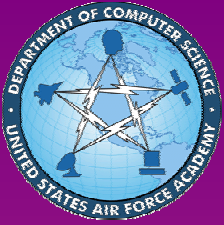
United States Air Force Academy

Colorado Springs, Colorado USA



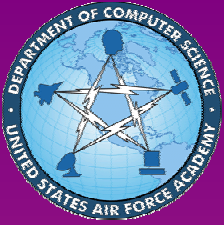
Outline

- An example video
- An improved OpenGL.ads
- Converting VRML models to Ada code
- Converting Ada code to C code
- Questions & Discussion



Outline

- An example video
- An improved OpenGL.ads
- Converting VRML models to Ada code
- Converting Ada code to C code
- Questions & Discussion



An Improved Opengl.ads

W. M. Richards Version (December 1997)

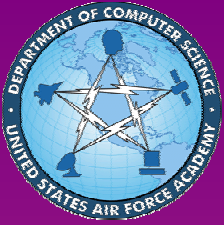
- OpenGL 1.1
- Strongly typed
- Example

```
procedure gLightf (light: LightIDEnum;  
                 pname: LightParameterEnum;  
                 param: GLfloat);
```

David Holm Version (2002-2003)

- OpenGL 1.3
- Weakly typed
- Example

```
procedure gLightf (light : GLenum;  
                 pname : GLenum;  
                 param : GLfloat);
```



An Improved Opengl.ads

Major modifications made to M. W. Richards' version

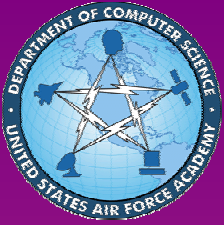
1. Changed basic data types to subtypes

- Old: `type GLfloat is new C.C_float;`
- New: `subtype GLfloat is C.C_float;`

2. Included new definitions for OpenGL versions 1.2, 1.3, 1.4

3. Reorganized the definitions to be consistent with the OpenGL Blue Book (Reference Manual)

4. General syntax cleanup

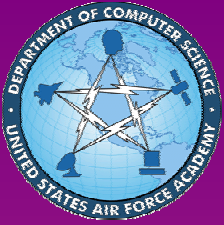


An Improved Opengl.ads

Major modifications made to M. W. Richards' version

5. Created explicit array types for points and vectors

- Old: `type GLfloatPtr is access all GLfloat;`
- New:
`type GLfloatPoint3 is array(0..2) of GLfloat;`
`type GLfloatPoint3Ptr is access all GLfloatpoint3;`



An Improved Opengl.ads

Issues during these modifications

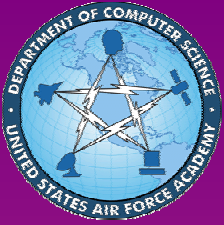
- I really missed a “*conditional include*” ability to facilitate version control.
- The requirement that enumerated types be defined in numerical order caused poorly organized definitions.



An Improved Opengl.ads

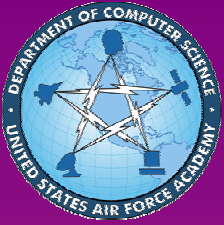
```
{define V1.1}

type TexturePaintingEnum is
(
  {V1.2}    GL_ADD,                {/V1.2}
  {V1.2}    GL_SRC_COLOR,         {/V1.2}
           GL_ONE_MINUS_SRC_COLOR,
           GL_SRC_ALPHA,
           GL_ONE_MINUS_SRC_ALPHA,
           GL_BLEND,
           GL_TEXTURE,
  {V1.1}    GL_REPLACE,          {/V1.1}
  {V1.3}    GL_MODULATE,         {/V1.3}
           GL_DECAL,
           GL_SUBTRACT,
           GL_COMBINE,
           GL_ADD_SIGNED,
           GL_INTERPOLATE,
           GL_CONSTANT,
           GL_PRIMARY_COLOR,
           GL_PREVIOUS
);
```

Outline

- An example video
- An improved OpenGL.ads
- Converting VRML models to Ada code
- Converting Ada code to C code
- Questions & Discussion

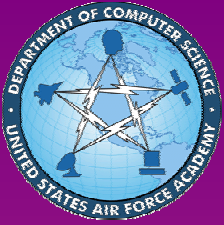


Converting VRML to Ada Code

VRML (Virtual Reality Modeling Language)

```
#VRML V1.0 ascii

Separator {
  DEF Sphere01 Separator {
    Material {
      diffuseColor 1.000 0.000 0.000
      ambientColor 0.100 0.000 0.000
    }
    Coordinate3 {
      point [ 0.000000 96.546875 -1.125000,
              0.000000 96.343750 0.812500,
              -0.375000 96.343750 0.781250,
              -0.734375 96.343750 0.671875,
              ...
            ]
    }
  }
}
```



Converting VRML to Ada Code

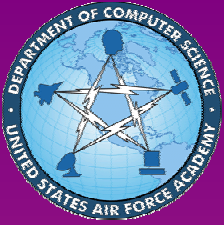
Generated Ada Code

```
Procedure Material3 is
```

```
    ambientColor   : array(1..3) of GLfloat := (0.1000, 0.0000, 0.0000);  
    diffuseColor   : array(1..3) of GLfloat := (1.0000, 0.0000, 1.0000);  
    specularColor  : array(1..3) of GLfloat := (1.0000, 1.0000, 1.0000);
```

```
begin
```

```
    glDisable(GL_COLOR_MATERIAL);  
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT,   ambientColor(1)'unrestricted_access);  
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE,   diffuseColor(1)'unrestricted_access);  
    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR,  specularColor(1)'unrestricted_access);  
    glMaterialf (GL_FRONT_AND_BACK, GL_SHININESS, 25.6 );  
end Material3;
```



Converting VRML to Ada Code

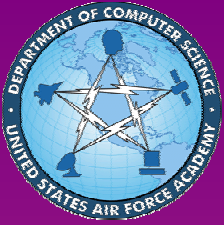
Major issue – scope of variables

```
-- Desired Code

Procedure draw_IndexedFaceSet1 is

  VertexArray : constant array(1..482*3) of GLfloat := (
    -0.3906250,  48.7343750,  -0.0468750,  --      0
    ...
  Triangles : constant array(1..960*3) of GLint := (
    0,    2,    1,  --      0
    0,    3,    2,  --      1
    ...
begin

  glEnableClientState(GL_VERTEX_ARRAY);
  glEnableClientState(GL_NORMAL_ARRAY);
  ...
end draw_IndexedFaceSet1;
```



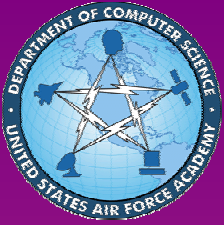
Converting VRML to Ada Code

Major issue – scope of variables

```
-- To avoid repeated instantiation

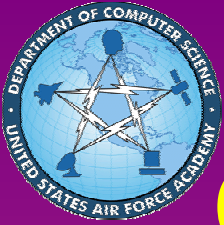
VertexArray1 : constant array(1..482*3) of GLfloat := (
    -0.3906250,  48.7343750,  -0.0468750,  --      0
    ...
Triangles1 : constant array(1..960*3) of GLint := (
    0,    2,    1,  --      0
    0,    3,    2,  --      1
    ...
Procedure draw_IndexedFaceSet1 is
begin

    glEnableClientState(GL_VERTEX_ARRAY);
    glEnableClientState(GL_NORMAL_ARRAY);
    ...
end draw_IndexedFaceSet1;
```



Outline

- An example video
- An improved OpenGL.ads
- Converting VRML models to Ada code
- Converting Ada code to C code
- Questions & Discussion



Converting Ada code to C code

- Easy tasks

- Converting statements, e.g.

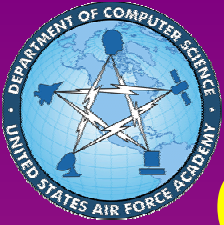
Ada *if expression then statements end if;*

C *if (expression) { statements }*

- Converting expressions, e.g. casts

Ada *float(x(3))*

C *(float)(x[2])*



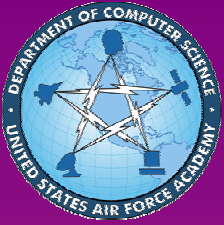
Converting Ada code to C code

- Difficult tasks
 - Creating correct case-sensitive identifiers (e.g., `glLight(...)`)
 - Dealing with arrays of arrays of arrays...
 - Converting array declarations
 - Dealing with “in” array parameters
 - Converting function calls that return non-scaler types



Converting Ada code to C code

- Issues not dealt with
 - Array slices
 - System libraries and calls (e.g., `Ada.Real_Time`)
 - Objects
 - References (aliases)
 - Dynamic memory allocation
 - Exceptions
 - Declare blocks
 - (many more)



Outline

- An example video
- An improved OpenGL.ads
- Converting VRML models to Ada code
- Converting Ada code to C code
- Questions & Discussion