UNIT15: STATIC MESH – MATERIAL SUBSET

【学習要項】
□Subsets
□Materials

【演習手順】
1．static_mesh クラスにサブセットを導入し、複数のマテリアル（テクスチャ）単位に描画できるよう機能拡張する
2．static_mesh クラスにメンバ変数を追加する
①サブセットは対応するマテリアル名、そのマテリアルを使用するメッシュのインデックス開始番号とインデックス数
※OBJ ファイルのキーワード usemtl に続く文字列がマテリアル名で、次の行以降に連続するキーワード f がその
マテリアルを使用するポリゴンであり、次のキーワード usemtl が出現するまで適用される

```
struct subset
{
  std::wstring usemtl;
  uint32_t index_start{ 0 };   // start position of index buffer
  uint32_t index_count{ 0 };   // number of vertices (indices)
};
std::vector<subset> subsets;
```

②MTL ファイルから取得するマテリアル情報構造体の定義
※テクスチャファイルネーム（texture_filename）を material 構造体のメンバ変数に変更する
※シェーダーリソースビュー（shader_resource_view）を material 構造体のメンバ変数に変更する
※MTL ファイルの仕様書（Mtl Specification.txt）を参照する
※今回は属性（Ka, Kd, Ks）は描画には使用しない

```
 1:  //std::wstring texture_filename;
 2:  //Microsoft::WRL::ComPtr<ID3D11ShaderResourceView> shader_resource_view;
 3:  struct material
 4:  {
 5:    std::wstring name;
 6:    DirectX::XMFLOAT4 Ka{ 0.2f, 0.2f, 0.2f, 1.0f };
 7:    DirectX::XMFLOAT4 Kd{ 0.8f, 0.8f, 0.8f, 1.0f };
 8:    DirectX::XMFLOAT4 Ks{ 1.0f, 1.0f, 1.0f, 1.0f };
 9:    std::wstring texture_filename;
10:    Microsoft::WRL::ComPtr<ID3D11ShaderResourceView> shader_resource_view;
11:  };
12:  std::vector<material> materials;
```

3．static_mesh クラスコンストラクタ

①OBJ ファイルパーサー部に下記コードを追加する
※サブセットのマテリアル名とインデックス開始番号のみ記録する

```
1:  else if (0 == wcscmp(command, L"usemtl"))
2:  {
3:    wchar_t usemtl[MAX_PATH]{ 0 };
4:    fin >> usemtl;
5:    subsets.push_back({ usemtl, static_cast<uint32_t>(indices.size()), 0 });
6:  }
```

②OBJ ファイルパーサー部終了後に下記コードを追加する
※サブセットのインデックス数を計算する

```
1:  std::vector<subset>::reverse_iterator iterator = subsets.rbegin();
2:  iterator->index_count = static_cast<uint32_t>(indices.size()) - iterator->index_start;
3:  for (iterator = subsets.rbegin() + 1; iterator != subsets.rend(); ++iterator)
4:  {
5:    iterator->index_count = (iterator - 1)->index_start - iterator->index_start;
6:  }
```

③MTL ファイルパーサー部に下記コードを追加する
※Kd と同様に Ka, Ks のデータも取得する

```
1:  if (0 == wcscmp(command, L"map_Kd"))
2:  {
3:      fin.ignore();
```

```
     wchar_t map_Kd[256];
     fin >> map_Kd;

     std::filesystem::path path(obj_filename);
     path.replace_filename(std::filesystem::path(map_Kd).filename());
*    //texture_filename = path;
*    materials.rbegin()->texture_filename = path;
*    fin.ignore(1024, L'¥n');
 }
*else if (0 == wcscmp(command, L"newmtl"))
*{
*    fin.ignore();
*    wchar_t newmtl[256];
*    material material;
*    fin >> newmtl;
*    material.name = newmtl;
*    materials.push_back(material);
*}
*else if (0 == wcscmp(command, L"Kd"))
*{
*    float r, g, b;
*    fin >> r >> g >> b;
*    materials.rbegin()->Kd = { r, g, b, 1 };
*    fin.ignore(1024, L'¥n');
*}
```

④すべてのマテリアルのテクスチャをロードし、シェーダーリソースビューを生成する

```
D3D11_TEXTURE2D_DESC texture2d_desc{};
//load_texture_from_file(device, texture_filename.c_str(),
//  shader_resource_view.GetAddressOf(), &texture2d_desc);
for (material& material : materials)
{
  load_texture_from_file(device, material.texture_filename.c_str(),
  material.shader_resource_view.GetAddressOf(), &texture2d_desc);
}
```

4．static_mesh クラスの描画関数をサブセット単位で描画するように変更する

```
void static_mesh::render(ID3D11DeviceContext* immediate_context,
  const XMFLOAT4X4& world, const XMFLOAT4& material_color)
{
  uint32_t stride{ sizeof(vertex) };
  uint32_t offset{ 0 };
  immediate_context->IASetVertexBuffers(0, 1, vertex_buffer.GetAddressOf(), &stride, &offset);
  immediate_context->IASetIndexBuffer(index_buffer.Get(), DXGI_FORMAT_R32_UINT, 0);
  immediate_context->IASetPrimitiveTopology(D3D11_PRIMITIVE_TOPOLOGY_TRIANGLELIST);
  immediate_context->IASetInputLayout(input_layout.Get());

  immediate_context->VSSetShader(vertex_shader.Get(), nullptr, 0);
  immediate_context->PSSetShader(pixel_shader.Get(), nullptr, 0);

  for (const material& material : materials)
  {
    immediate_context->PSSetShaderResources(0, 1, material.shader_resource_view.GetAddressOf());

    constants data{ world, material_color };
    XMStoreFloat4(&data.material_color, XMLoadFloat4(&material_color) * XMLoadFloat4(&material.Kd));
    immediate_context->UpdateSubresource(constant_buffer.Get(), 0, 0, &data, 0, 0);
    immediate_context->VSSetConstantBuffers(0, 1, constant_buffer.GetAddressOf());

    for (const subset& subset : subsets)
    {
      if (material.name == subset.usemtl)
      {
        immediate_context->DrawIndexed(subset.index_count, subset.index_start, 0);
```

```
28:         }
29:       }
30:     }
31:   }
```

５．static_mesh オブジェクト生成のコンストラクタ引数を.¥¥resources¥¥Mr.Incredible¥¥Mr.Incredible.obj に変更する

６．実行し、正しく描画されていることを確認する

７．他の obj ファイルをロード・描画テストする

【評価項目】
□メッシュのサブセット単位の描画