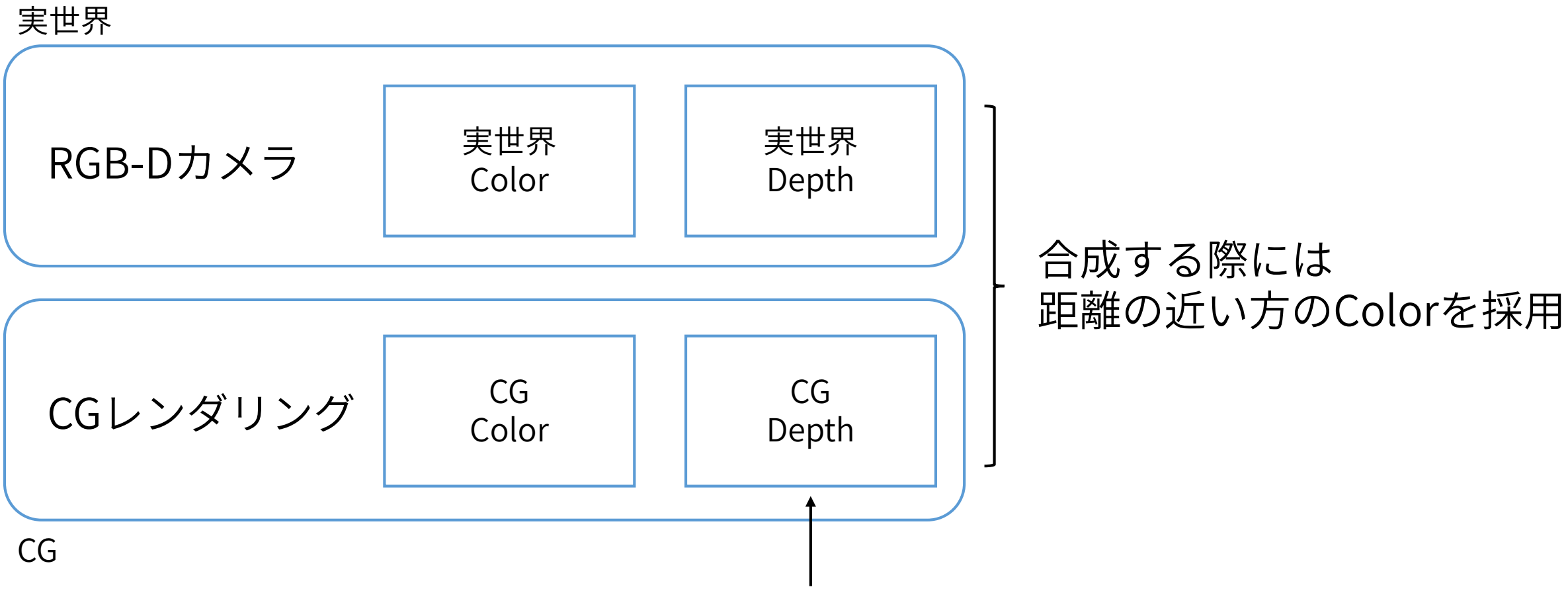
エルゴノミクスコンピューティング実習

CGとの合成

人間システム工学科 井村 誠孝
m.imura@kwansei.ac.jp

実世界とCGとの合成


実世界
RGB-Dカメラ
実世界
Color
実世界
Depth
CGレンダリング
CG
Color
CG
Depth
CG
合成する際には
距離の近い方のColorを採用
課題: CGの距離情報をどのようにして得るか?

CGの距離情報を得る: z-bufferの読み出し

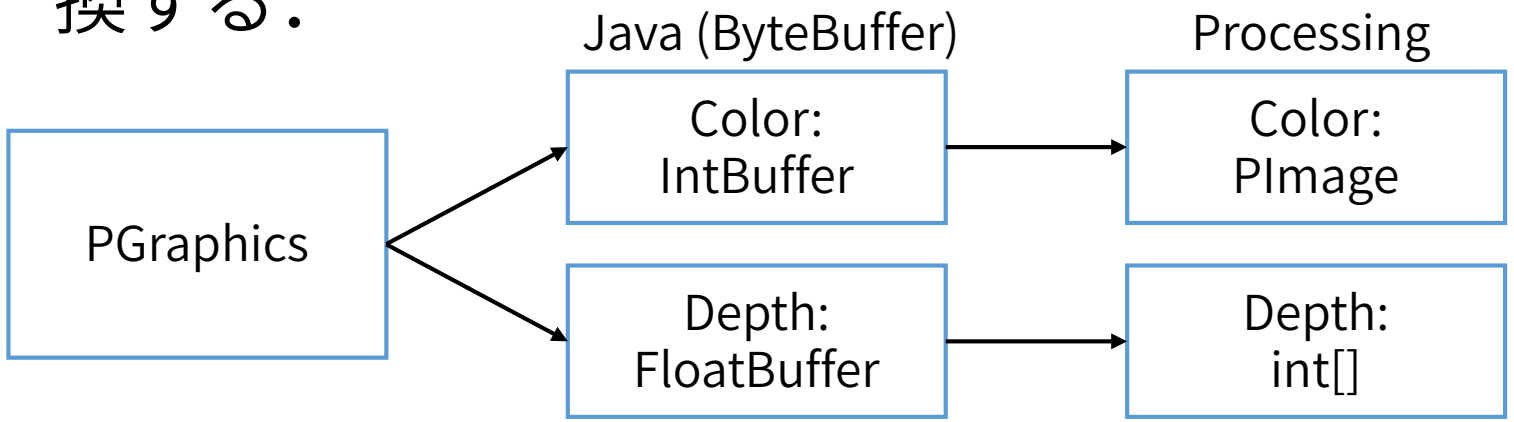
- コンピュータグラフィクスの描画にあたっては，隠面消去処理がなされる．
- 隠面消去処理にはいくつかの方法があるが，現在一般的に使われているのはz-buffer法
  - z-buffer(デプスバッファとも言う，以下デスプバッファで統一)と呼ばれる，各画素単位で距離情報(正確には，正規化座標系における奥行き情報)が格納されたバッファを準備する．
    - デスプバッファの大きさは画像と同じ
  - 描画前にデスプバッファをクリア(各距離を最も遠い距離に設定)
    - 一般的に最も近いと0，最も遠いと1．0と1に対応する距離はユーザが指定する．
  - 画素描画時に今から描画する情報が最前面であるか確かめ(デプステスト)，再前面であれば描画するとともに，デスプバッファを更新．
- デスプバッファの値を読み出すことで，RGB-Dカメラと同様の距離情報を得ることができる．

Processingでデプスバッファの値を得るには(1)

- 3次元レンダリング像を直接画面ではなく内部バッファに描く
  →PGraphicsクラスを用いる．
- PGraphicsを作成する．
  PGraphics PG;
  pg = createGraphics(IMAGE_WIDTH, IMAGE_HEIGHT, P3D);
- 各種の描画コマンドの先頭に，「pg.」を付けて実行すると，画面ではなくpgに描画される．
- 描画された内容を画面に表示するには，image()関数を用いる．
  - PGraphicsはPImageを継承しているので，PImageと同じように扱える．

Processingでデプスバッファの値を得るには(2)

- PGraphicsが内部に保持しているデプスバッファの値を得る．
- ProcessingのP3Dモードは，内部でOpenGLを利用しているため，OpenGLの機能を用いる．
  - Javaの機能なのでProcessingのマニュアルには記述されていない．
- Javaでバイト配列の操作を行うByteBufferクラス(正確にはそれを継承したIntBufferやFloatBuffer)にPGraphicsから情報を読み出す．
- ByteBufferに格納された値を，Processing標準のデータ形式に変換する．



Processingでデプスバッファの値を得るには(3)

- デプスバッファに格納されている値は，最も近い=0，最も遠い=1に正規化されている．
- 3DCGではレンダリングの際に，距離によるクリッピング(ある距離範囲内のみレンダリング)が行われる．カメラの設定でレンダリング距離範囲を指定する．
  - 通常，near clippingとfar clippingと呼ばれる，いずれも正の値．
  - pg.perspective(縦画角[rad], アスペクト比, near clipping, far clipping);
- デプスバッファの値dと，距離zとの間には，以下の関係がある．

$$z = \frac{2NF}{F + N - (2d - 1)(F - N)}$$

  - N: near clipping
  - F: far clipping

以上のサンプルのソースコードを確認する．

関連部分のソースコード(1) – importと宣言

```
import java.nio.*;

import com.jogamp.opengl.GL;
import com.jogamp.opengl.GL2ES2;

PGraphics pg;
IntBuffer pgColor;
FloatBuffer pgDepth;

PImage virtualColorImage;
int[] virtualDepthBuffer;
```

必要なライブラリをimport

各種宣言

関連部分のソースコード(2) – バッファの確保

```
pgColor = ByteBuffer.allocateDirect((IMAGE_WIDTH * IMAGE_HEIGHT) << 2)
          .order(ByteOrder.nativeOrder()).asIntBuffer();
pgDepth = ByteBuffer.allocateDirect((IMAGE_WIDTH * IMAGE_HEIGHT) << 2)
          .order(ByteOrder.nativeOrder()).asFloatBuffer();
```

ByteBufferの確保

関連部分のソースコード(3) – バッファへの読み出し

```
PGraphics3D pg3d = (PGraphics3D) pg;
FrameBuffer fb = pg3d.getFrameBuffer();

PJOGL pgl = (PJOGL) beginPGL();
GL2ES2 gl = pgl.gl.getGL2ES2();

gl.glBindFramebuffer(GL.GL_READ_FRAMEBUFFER, fb.glFbo);
gl.glReadPixels(0, 0, IMAGE_WIDTH, IMAGE_HEIGHT, GL.GL_BGRA, GL.GL_UNSIGNED_BYTE, pgColor);
gl.glReadPixels(0, 0, IMAGE_WIDTH, IMAGE_HEIGHT, PGL.DEPTH_COMPONENT, GL.GL_FLOAT, pgDepth);

endPGL();
```

OpenGLの関数を使用するための準備

色情報とデプス情報をByteBufferに読み出す

関連部分のソースコード(4) – Processing用に変換

```
virtualColorImage = createImage(IMAGE_WIDTH, IMAGE_HEIGHT, RGB);
virtualColorImage.loadPixels();
for (int i = 0; i < virtualColorImage.pixels.length; i++) {
  virtualColorImage.pixels[i] = pgColor.get(i);
}
virtualColorImage.updatePixels();

for (int y = 0; y < IMAGE_HEIGHT; y++) {
  for (int x = 0; x < IMAGE_WIDTH; x++) {
    float d = pgDepth.get(x + y * IMAGE_WIDTH);
    d = 2 * d - 1;
    float z = 2.0 * NEAR * FAR / (FAR + NEAR - d * (FAR - NEAR));
    virtualDepthBuffer[x + y * IMAGE_WIDTH] = int(z);
  }
}
```

色情報をPImageにセット

デプス情報をPImageにセット

0-1の値を距離に変換

実世界とCGとの合成

- 実世界のカメラの設定と，CGレンダリング時のカメラの設定を，一致させる必要がある．
- 本来は外部パラメータ(位置姿勢情報)も一致させるべきであるが，本実習では内部パラメータ(画角など)の一致のみ行う．すなわちレンダリング時のカメラの画角をIntel RealSenseのデプスカメラと同じに設定する．
- またIntel RealSenseのカラーカメラとデプスカメラは取り付け位置が異なることから，デプス画像代わりにカラー画像を提示するためにはカメラ配置とデプス値を考慮した画像間の座標変換が必要である．
  Processingのライブラリはオフィシャルなものではないのでこの機能が準備されていない．自力でカメラキャリブレーションを行えば実現可能であるが，今回の実習では行わない．

本日の演習

- 実空間のデプス画像に，CG画像を合成する
  - テンプレートのソースコードでのCG空間の設定: カメラから1m離れた点を中心に，半径0.5mで球体が回転している．
  - カメラからおおよそ1m離れた場所に立ったときに，球体が自分の周囲をまわっているように見えるよう，隠蔽関係を適切に判断して，画像を合成せよ．
- [発展] 実空間距離画像から人を検出し(まじめにやるとたいへんなので，ある一定の距離範囲内の，最も近い点など)，その人の周囲を球体が回転するようにせよ．
- [発展] その他，実空間の距離情報を用いて，バーチャル空間(CG)とインタラクションするものを独自に考案してもよい．
- テンプレートがあります．
- ソースコード，および，レポート(正しく隠蔽関係が描画されているスクリーンショットを含む)を提出してもらいます．

ソースコードの重要な部分

- 実空間の情報は，以下に格納されている．
  - 色: PImage realColorImage;
  - デプス: int[] realDepthBuffer;
- バーチャル空間(CG)の情報は，以下に格納されている．
  - 色: PImage virtualColorImage;
  - デプス: int[] virtualDepthBuffer;
- テンプレートでは，以下に色画像およびデプス画像を格納すると，ウィンドウに表示されるようになっている．
  - PImage colorImage;
  - PImage depthImage;