エルゴノミクスコンピューティング実習

03 基礎知識 - 3D座標変換

人間システム工学科 井村 誠孝
m.imura@kwansei.ac.jp

http://j.mp/kg-ergo-p

サンプル: 3D用のウィンドウを開くだけ

```
void setup() {
  size(512, 512, P3D);
  frameRate(60);
}

void draw() {
  background(255);
}
```

関数 setup()
実行開始後に一度だけ実行

size(x, y, P3D)
ウィンドウサイズを指定
OpenGLを用いた3D CG
レンダラを指定

frameRate(f)
フレームレートを指定

関数 draw()
繰り返し実行される

background(c)
背景を指定色でクリア

カメラの設定

- デフォルトだと，2Dの場合と同じく，画面左上が原点になる．扱いにくい．
- 関数 camera() を用いてカメラの設定をすることで，例えば座標の原点を画面中心にできる．
  - (※) 第7-9引数は上方向ベクトルではなくて下方向?

```
void draw() {
  background(255);
  camera(50, 40, 30, 0, 0, 0, 0, 0, -1);
}
```

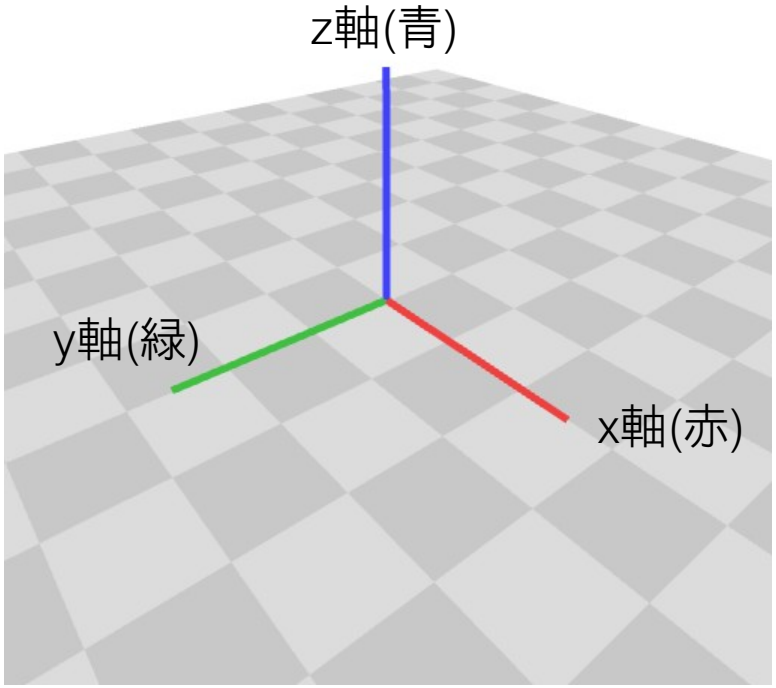
注視点の位置 (x, y, z)

視点の位置 (x, y, z)

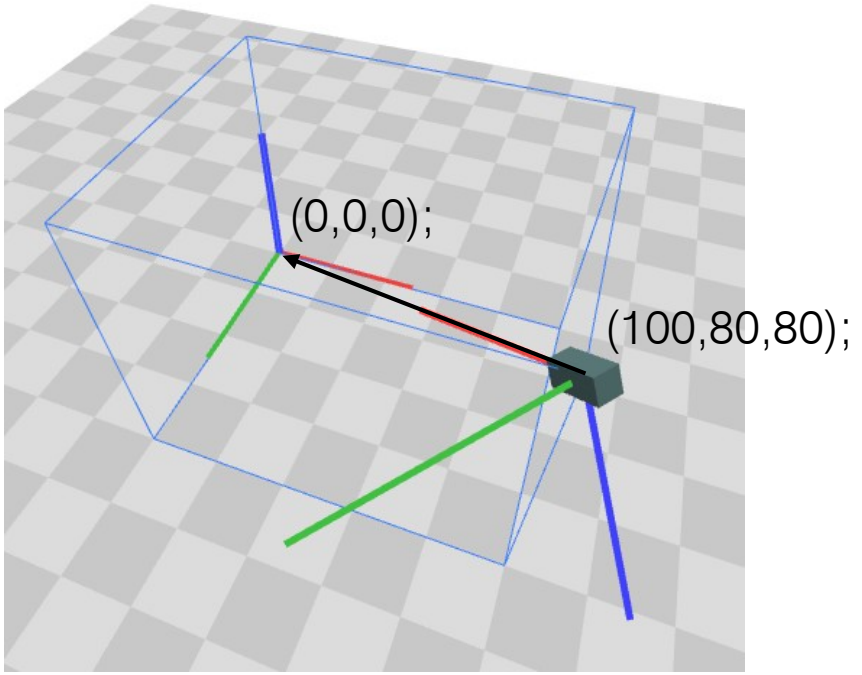
カメラ上方向のベクトル(※)

カメラの設定

```
camera(100, 80, 80, 0, 0, 0, 0, 0, -1);
```



カメラの視界



カメラ配置は実際こうなってます

3次元での物体の描画

- 先週実施した2次元での描画と同様に，カレント座標系を適切に移動させながら，描画を行っていく．
- Processingで用意されている3次元物体は以下の通り．
  - box() - 直方体を描く
  - sphere() - 球体を描く
  - Pshapeクラスを用いると3次元モデルを扱える．
    - loadShape()でobj形式の3次元モデルを読み込むことができる．

カレント座標系の変換

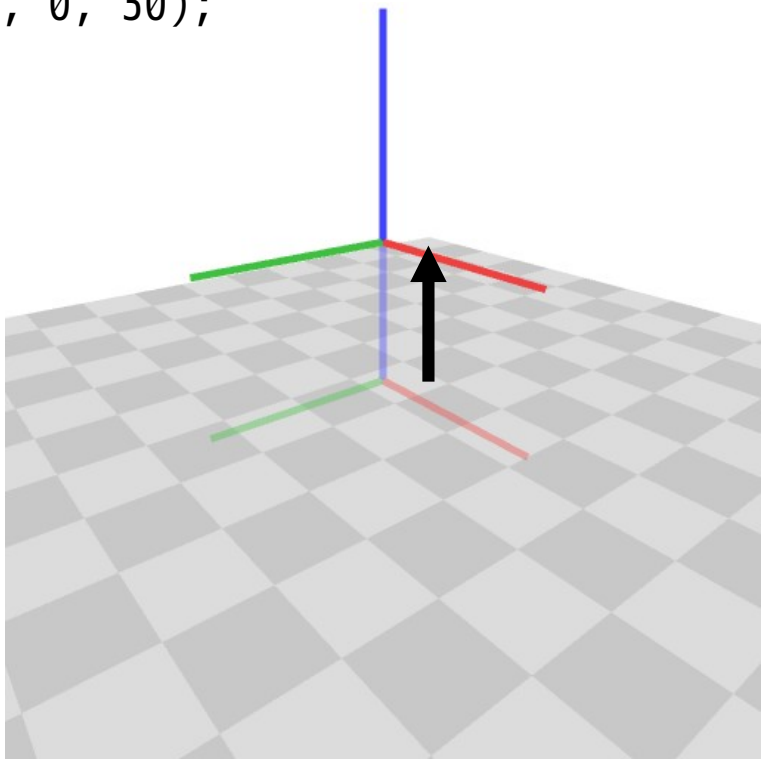
- 平行移動(並進): `translate(x, y, z)`
- 回転(X軸まわり): `rotateX(theta)`
- 回転(Y軸まわり): `rotateY(theta)`
- 回転(Z軸まわり): `rotateZ(theta)`
- 拡大縮小: `scale(s)`

- いずれもカレント座標系に対して適用される
- 回転の角度はラジアン
- Processingの座標系は左手系(なぜに…)

平行移動(並進)

- `translate(x, y, z)`
  - カレント座標系を (x, y, z) だけ移動させる.

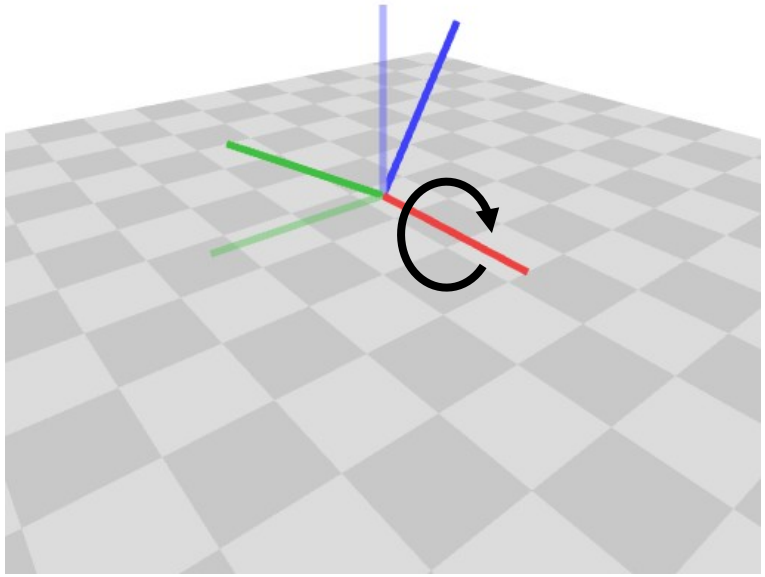
例: `translate(0, 0, 30);`



回転(X軸まわり)

- `rotateX(theta)`
  - カレント座標系をX軸まわりにthetaだけ回転させる.

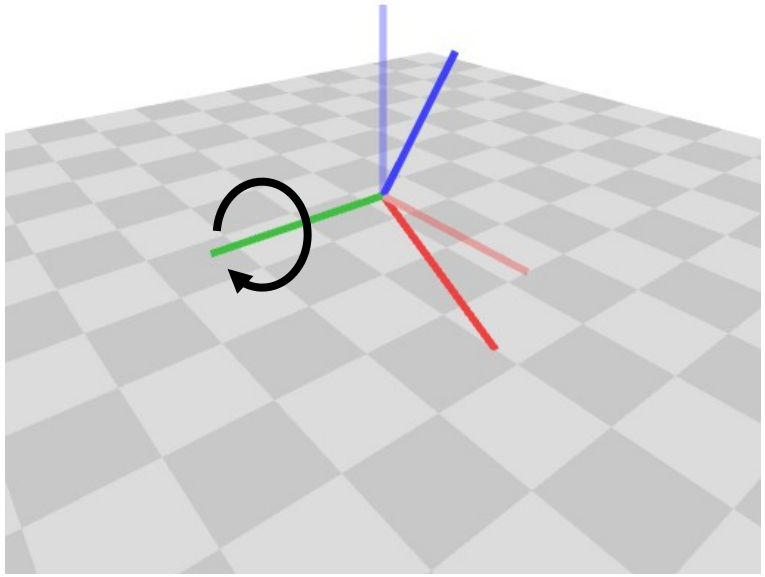
例: `rotateX(PI / 6);`



回転(Y軸まわり)

- `rotateY(theta)`
  - カレント座標系をY軸まわりにthetaだけ回転させる.

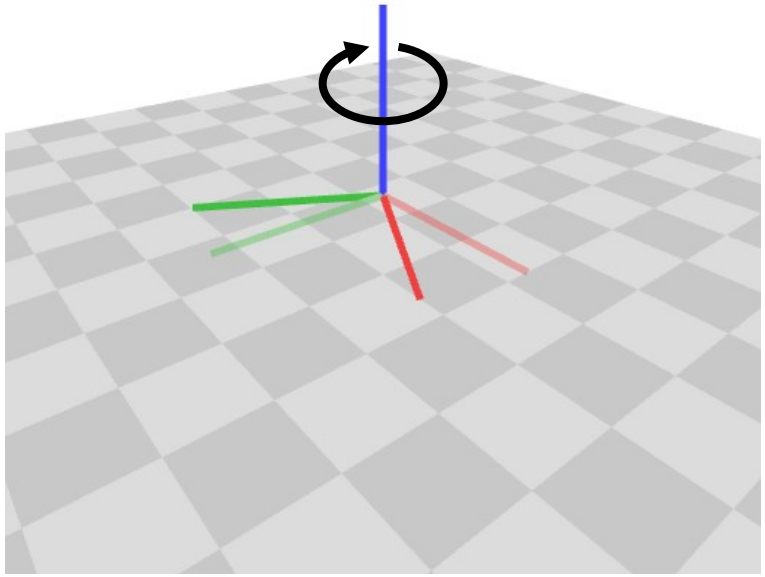
例: `rotateY(PI / 6);`



回転(Z軸まわり)

- `rotateZ(theta)`
  - カレント座標系をZ軸まわりにthetaだけ回転させる.

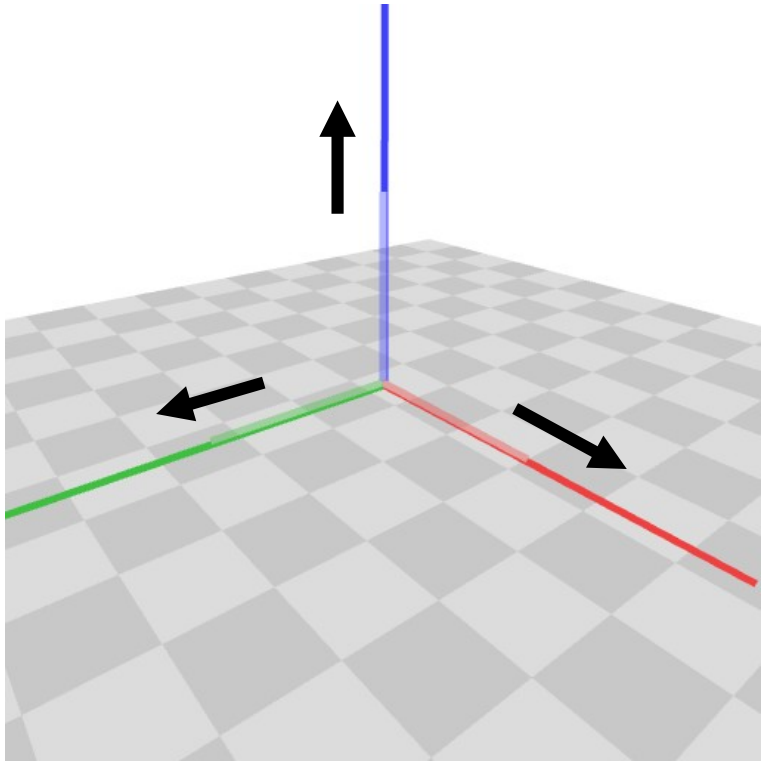
例: `rotateZ(PI / 6);`



拡大縮小

- scale(s)
  - カレント座標系をs倍に拡大(縮小)する.

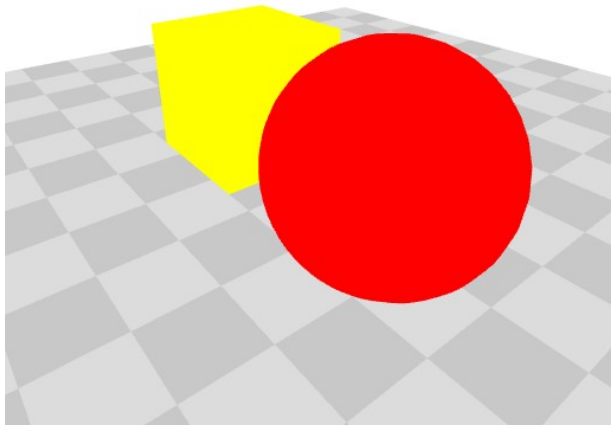
例: scale(2.0);



ライティング

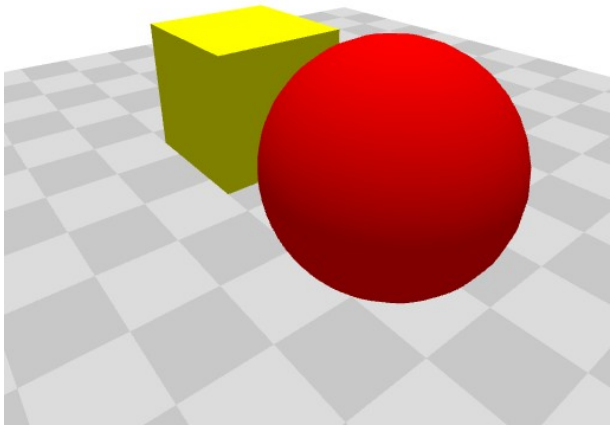
- draw()の最初で関数 lights() を実行すると，デフォルトの光源が設定される．
  - 環境光: RGB＝(128, 128, 128)
  - 平行光源: RGB＝(128, 128, 128)，光の向き＝(0, 0, -1)，減衰なし

noLights()



lights()

光の向き



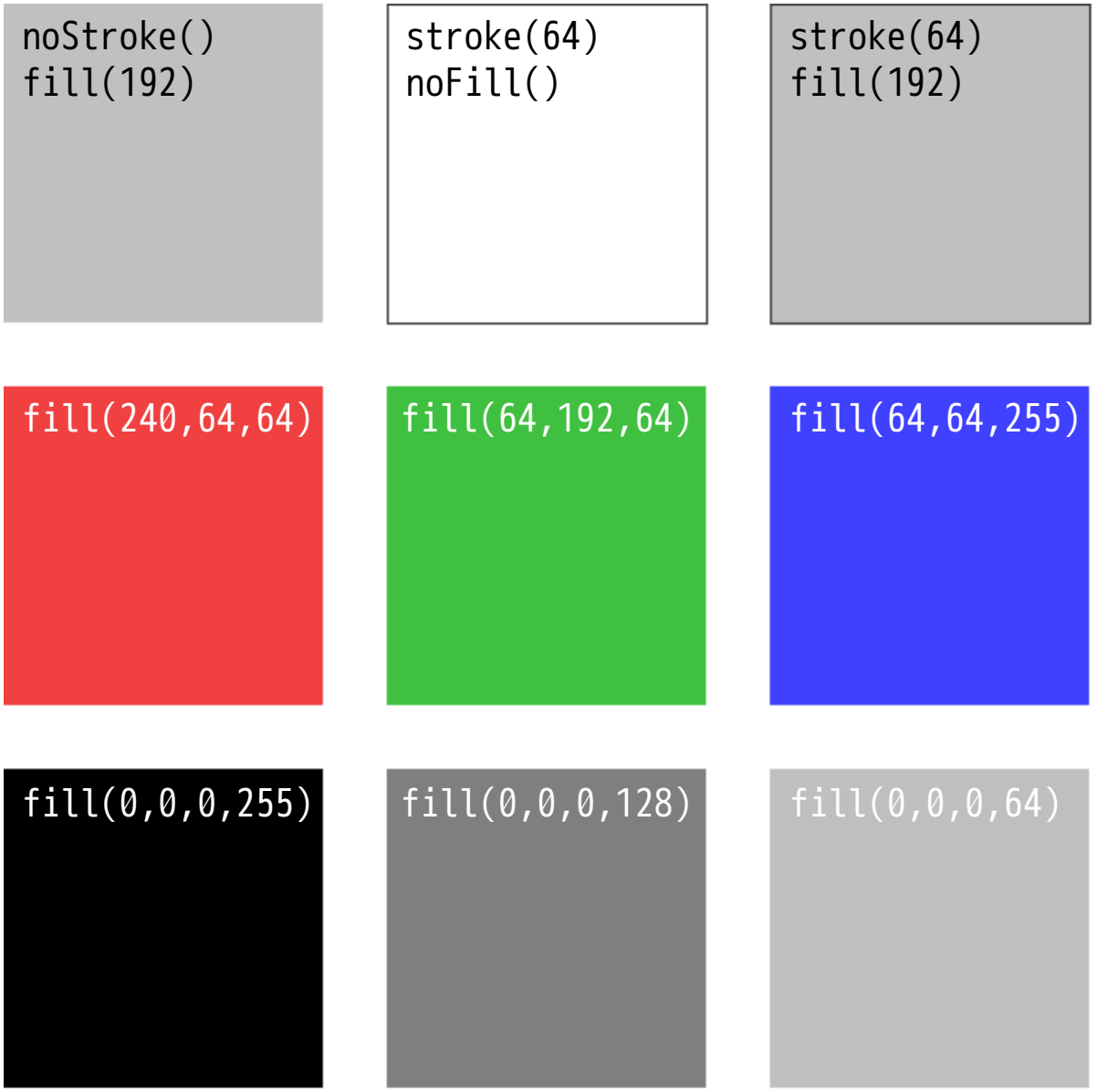
描画色の指定(再掲)

- 数字1個: グレイスケール
- 数字3個: RGB
  - R: 赤
  - G: 緑
  - B: 青
- 数字4個: RGBA
  - A: 透明度 (アルファチャンネル)

- 値の範囲はデフォルトでは0-255. 関数 `colorMode()` で変更可能.
- HSB(HSV)にも切り替え可能

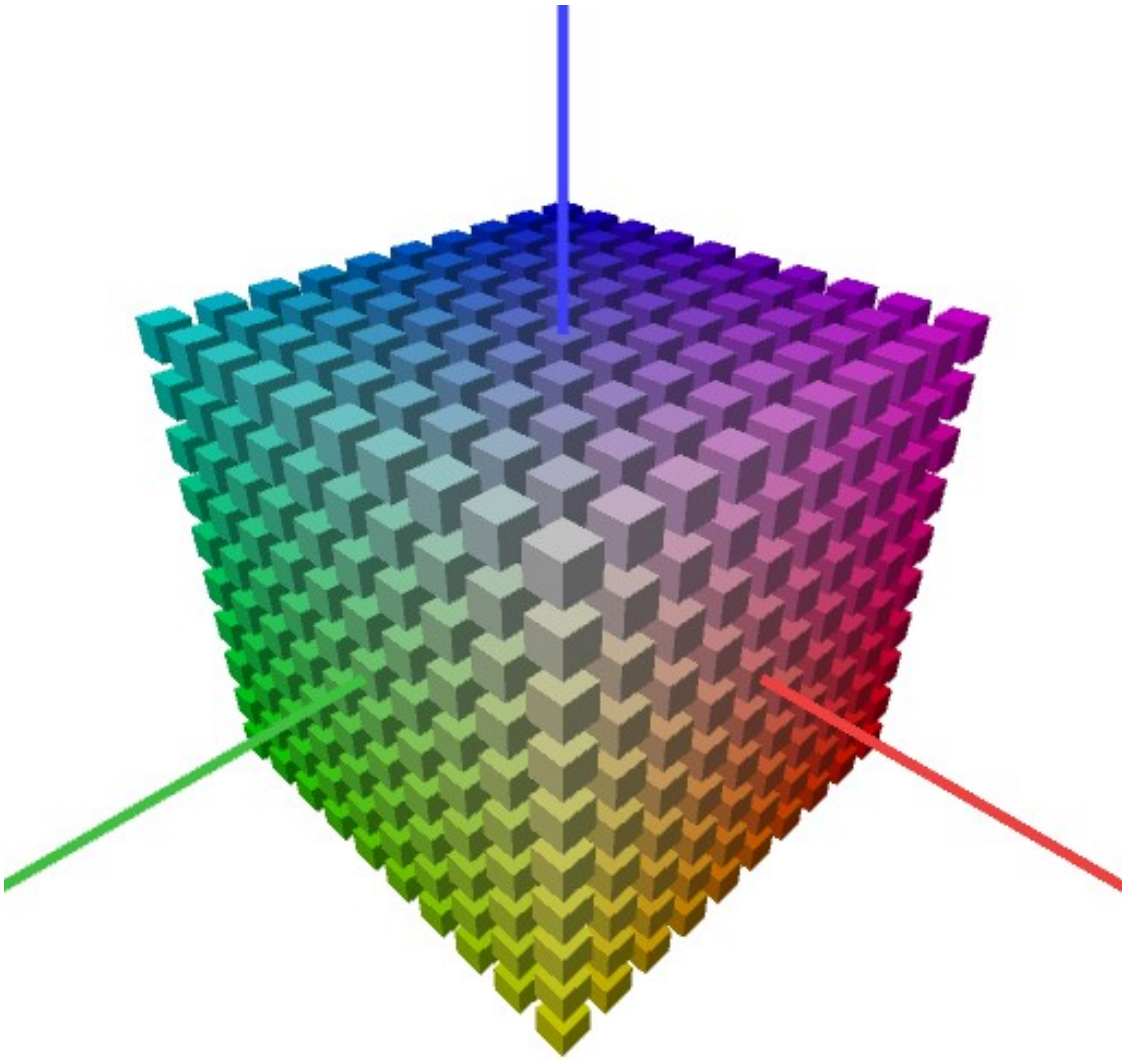
- 線の色: `stroke(c)`
  - 線を描きたくない場合は `noStroke()`
- 塗りの色: `fill(c)`
  - 塗り潰したくない場合は `noFill()`

色の指定 サンプル


noStroke()
fill(192)
stroke(64)
noFill()
stroke(64)
fill(192)
fill(240,64,64)
fill(64,192,64)
fill(64,64,255)
fill(0,0,0,255)
fill(0,0,0,128)
fill(0,0,0,64)

http://j.mp/kg-ergo-p

RGB色空間の描画例



サンプル: color_cube

アニメーション

- 物体をアニメーションさせたい.
- draw() の中で複数フレームの描画はできない.
- 関数 millis() でプログラム開始時からの経過時間をミリ秒単位で取得できるので，これを使ってアニメーションに必要なパラメータを計算する.
- 例: 周期 PERIOD (単位:ミリ秒)で物体を回転させる.

```
float PERIOD = 3 * 1000;
float theta = TWO_PI * millis() / PERIOD;
rotateZ(theta);
box(20);
```

サンプル: animation_sample