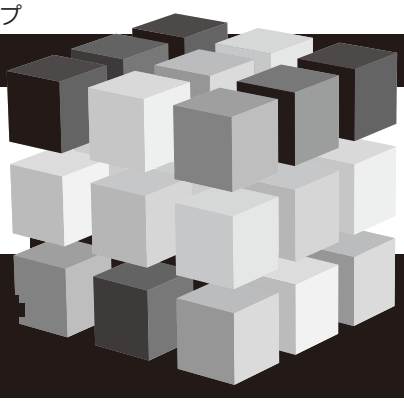
ダウンロード・データあります

画像処理プログラムを例に…
最適化による処理高速化にトライ

ラズパイ×OpenCL GPUアプリ作り



最終回
第3回　RGB/2値変換アプリの作成

本橋 弘臣

リスト1　カーネル・コードの後段に追加するコード
2通りの方法でグレースケール/2値変換処理を実装

```
// THRESH: 多値画像データを2値化する際のしきい値

#if 1
  // 画像2値化コード(三項演算子)

  // グレースケールデータを2値化する.
  gs1 = (gs1 > THRESH) ? 255 : 0;
  gs2 = (gs2 > THRESH) ? 255 : 0;
  gs3 = (gs3 > THRESH) ? 255 : 0;
  gs4 = (gs4 > THRESH) ? 255 : 0;

  // 2値データをメモリに書き込む.
  buff_gs[off_gs] = (uchar4)(gs1, gs2, gs3, gs4);
#else
  // 画像2値化コード(if文)

  // グレースケールデータを2値化する.
  out_u8x4 = (uchar4)(0);
  if (gs1 > THRESH) out_u8x4.s0 = 255;
  if (gs2 > THRESH) out_u8x4.s1 = 255;
  if (gs3 > THRESH) out_u8x4.s2 = 255;
  if (gs4 > THRESH) out_u8x4.s3 = 255;

  // 2値データをメモリに書き込む.
  buff_gs[off_gs] = out_u8x4;
#endif
```

三項演算子を利用

if文を利用

前回(第2回，2025年9月号)説明したRGB/グレースケール画像変換プログラム(opencl-rgb2gs-opt2)をベースに，入力したフルカラー画像ファイルを白黒2値のグレースケール画像に変換するOpenCLアプリケーションを作成します．RGB/2値画像に変換する処理自体は，次の処理ステップで実現できます．

- (1) RGBフルカラー画像をグレースケール画像に変換
- (2) グレースケール画像の各画素の画素値が127を超える場合は画素値を255に，127以下の場合は画素値を0に変更

今回のようにカーネルでの画像処理内容のみを変更する場合には(カーネル関数の1回実行ごとの仕事量やカーネル関数を実行する回数に変更がない場合)，ホスト・コード側では一切の変更を行う必要がありません．

表1　変換処理方式による処理速度の違い
三項演算子("?")を利用した方が速い

カーネル処理データ・サイズ	グレースケール/2値変換方式	測定結果	
		処理時間	処理速度
uchar4x3	三項演算子	6.2ms	3.2Gバイト/s
uchar4x3	if文	7.1ms	2.8Gバイト/s

2値変換用カーネル・コードにするための追加点

カーネル・コードに関しては，opencl-rgb2gs-opt2プログラムにおいてRGB/グレースケール変換処理をカーネル・コードに実装済みです．その処理の後段にグレースケール/2値変換処理を追加します．**リスト1**にカーネル・コードに対する追加部分のコードを示します．

グレースケール/2値変換処理を実装するためには，入力画素値に応じて出力画素値を切り替える必要があるわけですから，何らかの方法で条件分岐的な処理を実現する必要があります．今回はコードの書き方によって処理性能にどのような影響が出るのかを比較するために，次の2通りの方法でグレースケール/2値変換処理を実装してみました．

- 三項演算子("?")を利用する
- if文を利用する

この2通りの実装方法について，カーネル・コードの性能測定を実施した結果を**表1**にまとめました．今回の測定は分岐処理の書き方と処理性能の関係を調査することを目的としていましたので，ローカル・ワーク・サイズは(256，1)で固定しています．

この測定結果から分かるように，if文を使用するよりも三項演算子を利用した方が速いということが分かりました．それではなぜGPUでif文を使用すると遅くなるのでしょうか．実を言うと，GPUは分岐処理が大変苦手なのですが，そのことにはちゃんと理由があります．