

計算リソースはCPUだけじゃない!
画像処理の高速化や演算処理のオフロードに

ラズベリー・パイで始める GPUプログラミング

本橋 弘臣

第2回 初めてのGPUプログラム実行

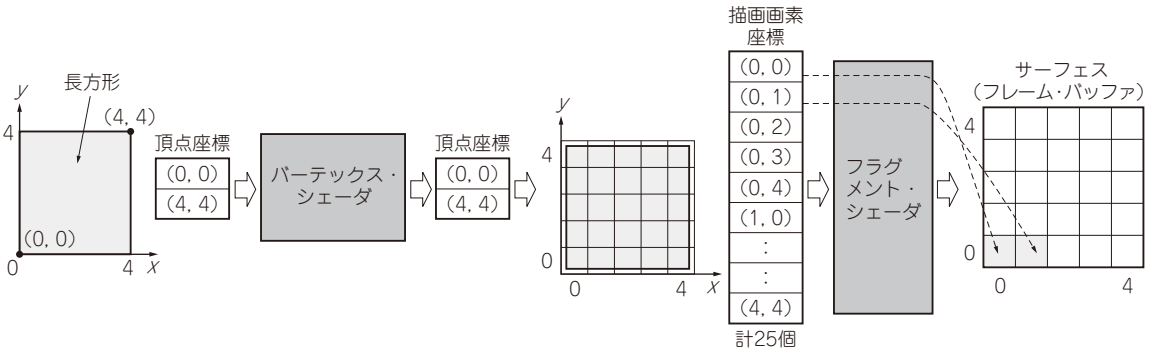


図1 OpenGLでグラフィックス描画処理を行う流れ
バーテックス・シェーダで移動/回転させ、フラグメント・シェーダで実際の画素値を1個ずつ出力する

本連載では、ラズベリー・パイ5を例に、GPUプログラミングの基礎知識とコーディング手法を紹介します。

今回は、実際にGPUで動作するGPGPUプログラムを作成します。具体的には、メインCPU側で動作するプログラム(ホスト・コード)と、GPU側で動作するプログラム(デバイス・コード)の両方を作成します。

実際にプログラムを作成する前に、OpenGLでの画面描画の仕組みやGPGPU処理の流れについて、模擬プログラムを交えて解説します。(編集部)

OpenGLでの画面描画の仕組み

● グラフィックス描画の最小単位

3Dグラフィックス・プログラミングにおける描画の最小単位は三角形です。この三角形を3次元空間中に多数配置することで、3Dゲームに登場する人物や物体を表現しています。3D対応のGPUを利用して2D描画を行う場合は、三角形頂点のZ座標を全て0に固定します。

● グラフィックス描画処理の流れ

図1に長方形1個を2D描画する場合のグラフィッ

クス描画処理の流れを示します。

グラフィックス描画の最小単位は三角形なのですが、ここでは説明をシンプルにするため、長方形の描画に対応したGPUを使っているものと仮定します。ここで、座標(0, 0)-(4, 4)の範囲に囲まれる長方形を描画する場合には、バーテックス・シェーダに入力する頂点座標として2つの座標値を用意します。

PCでグラフィックス描画を行う際の座標の原点(0, 0)は、一般的には画面の左上ですが、OpenGLの原点は左下です。

▶ステップ①…バーテックス・シェーダ

3Dグラフィックス描画を行うときは、バーテックス・シェーダに入力された頂点座標に対して、3次元空間中で移動/回転させるために行列演算を行います。ただ、2Dグラフィックス描画を行う場合や、今回のようにOpenGL ES (GLES)でGPGPU演算を行う場合は、バーテックス・シェーダでは特に何も演算を行わず、入力された頂点をそのまま出力します。

▶ステップ②…フラグメント・シェーダ

図1には示していませんが、フラグメント・シェーダの前段回路では、バーテックス・シェーダが出力した頂点座標を、全画素分の描画画素座標に変換する処理が行われています。この描画画素座標がフラグメント・シェーダへの入力となります。