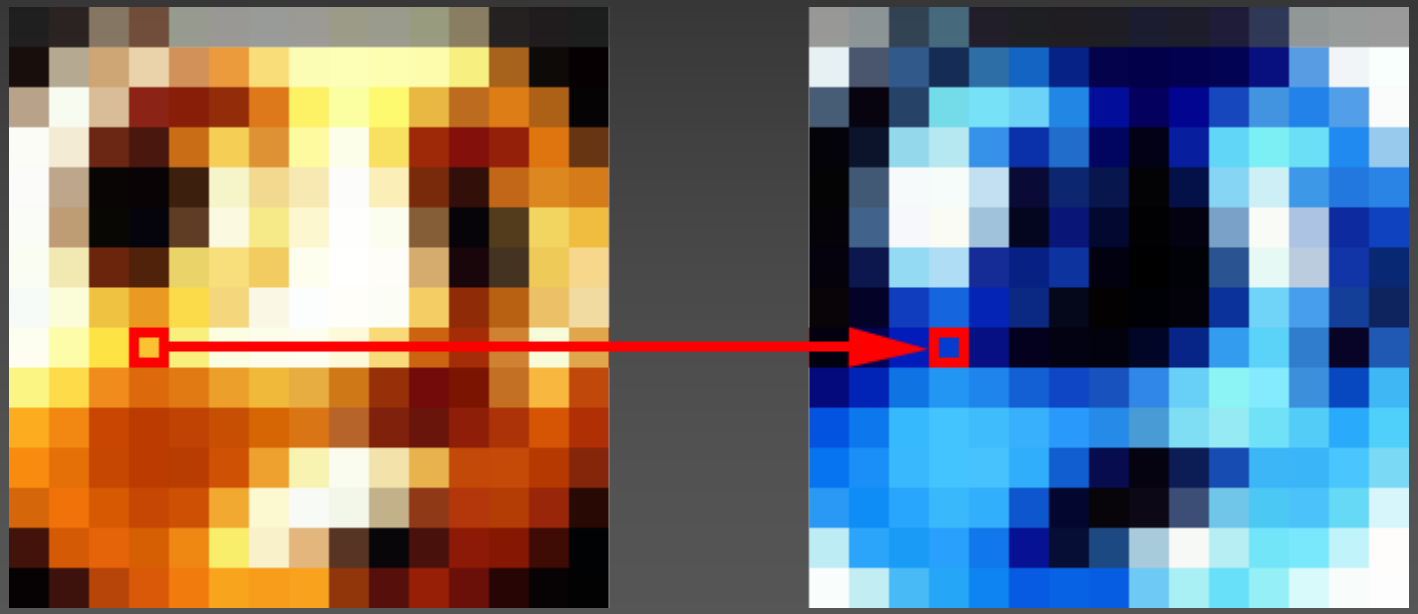
おしゃれ

Flex 4 Pixel Shader Effects

taiga.jp
廣畑 大雅

Pixel Shader とは

- ピクセルを操作する機能



© 2010 taiga.jp

Flex 4 Pixel Shader Effects

- spark.effects.Wipe
  - ✓ WipeUp.pbk
  - ✓ WipeUp.pbj
  - ✓ WipeDown.pbk
  - ✓ WipeDown.pbj
  - ✓ WipeRight.pbk
  - ✓ WipeRight.pbj
  - ✓ WipeLeft.pbk
  - ✓ WipeLeft.pbj

© 2010 taiga.jp

Flex 4 Pixel Shader Effects

- spark.effects.CrossFade
  - ✓ CorssFade.pbk
  - ✓ CrossFade.pbj

© 2010 taiga.jp

spark.effects.Wipe

© 2010 taiga.jp

spark.effects.CrossFade

© 2010 taiga.jp

Shader Effect の使い方 1

State と Transition を使用する場合

© 2010 taiga.jp

State の定義

```
<s:states>
    <s:State name="first" />
    <s:State name="second" />
    <s:State name="third" />
</s:states>
```

© 2010 taiga.jp

エフェクト対象の定義

```
<s:BitmapImage
    id = "image"
    source.first = "@Embed(...)"
    source.second = "@Embed(...)"
    source.third = "@Embed(...)"
 />
```

© 2010 taiga.jp

Transition とエフェクトの定義

```
<s:transitions>
    <s:Transition fromState="*" toState="*">
        <s:CrossFade
            target = "{image}"
        />
    </s:Transition>
</s:transitions>
```

© 2010 taiga.jp

遷移処理

```
switch(index) {
    case 0: currentState = "first";  break;
    case 1: currentState = "second"; break;
    case 2: currentState = "third";  break;
}
```

© 2010 taiga.jp

Shader Effect の使い方 2

ViewStack を使用する場合

© 2010 taiga.jp

ViewStack の定義

```
<mx:ViewStack
    id = "viewStack"
    change = "changeHandler(event)"
    creationPolicy = "all"
>

    ...

</mx:ViewStack>
```

© 2010 taiga.jp

エフェクトの定義

```
<fx:Declarations>
    <s:CrossFade
        id = "effect"
        target = "{viewStack}"
    />
</fx:Declarations>
```

© 2010 taiga.jp

続・エフェクトの定義と遷移処理

```
protected function setIndexAt(v:int):void {
    var index:int = viewStack.selectedIndex;
    var target:IUIComponent =
    viewStack.getChildAt(index) as IUIComponent;

    effect.bitmapFrom =
    BitmapUtil.getSnapshot(target);

    viewStack.selectedIndex = v;
}
```

© 2010 taiga.jp

change イベントハンドラ処理

```
protected function
  changeHandler(event:IndexChangedEvent):void

{

    callLater(effect.play, []);

}
```

© 2010 taiga.jp

Pixel Shader Effect クラスを自作

- Pixel Bender Toolkit を使用して PBK を書く

注：PBK (Pixel Bender kernel language)
↓
ピクセルシェーダ

© 2010 taiga.jp

または PBDT が便利

- PBDT
  http://blog.joa-ebert.com/pbdt/
  (FDT と連携で .pbk -> .pbj 自動書き出し)

- update site
  http://pbdt.joa-ebert.com/update

© 2010 taiga.jp

最小の PBK

```
<languageVersion: 1.0;>
kernel NewKernel
<
    namespace: "taiga";
    vendor: "taiga.jp";
    version: 1;
    description: "Your Description";
>
{
    input image4 src0;
    output pixel4 dst;
    void evaluatePixel() {
        dst = sampleNearest(src0, outCoord());
    }
}
```

© 2010 taiga.jp

画素数（ピクセルの個数）回 実行

```
<languageVersion: 1.0;>
kernel NewKernel
<
    namespace: "taiga";
    vendor: "taiga.jp";
    version: 1;
    description: "Your Description";
>
{
    input image4 src0;
    output pixel4 dst;
    void evaluatePixel() {
        dst = sampleNearest(src0, outCoord());
    }
}
```

© 2010 taiga.jp

必須パラメータ

- Flex 4 のエフェクトとして使用するときは、下記予約変数が必須なので注意

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
```

© 2010 taiga.jp

progress

- AnimateTransitionShaderInstance クラスで予約されている 0.0~1.0 の進捗パラメータ

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
```

© 2010 taiga.jp

width, height

- AnimateTransitionShaderInstance クラスで予約されている幅と高さのパラメータ

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
```

© 2010 taiga.jp

src0

- 標準で使用される image4 型の第 1 入力
(名前は任意で OK)

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
```

© 2010 taiga.jp

NG の例

- 第 1 入力であることが必須

```
parameter float progress;
parameter float width;
parameter float height;
input image4 from;
input image4 to;
input image4 src0;
output pixel4 dst;
```

© 2010 taiga.jp

from, to

- AnimateTransitionShaderInstance クラスで予約されているエフェクト適応前と後の入力 (BitmapData)

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
```

© 2010 taiga.jp

dst

- 標準で使用される pixel4 型の出力
(名前は任意で OK)

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
```

© 2010 taiga.jp

必須処理

- 標準入力をサンプリングして任意の変数に代入

```
parameter float progress;
parameter float width;
parameter float height;
input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
void evaluatePixel() {
    dst = sampleNearest(src0, outCoord());
}
```

© 2010 taiga.jp

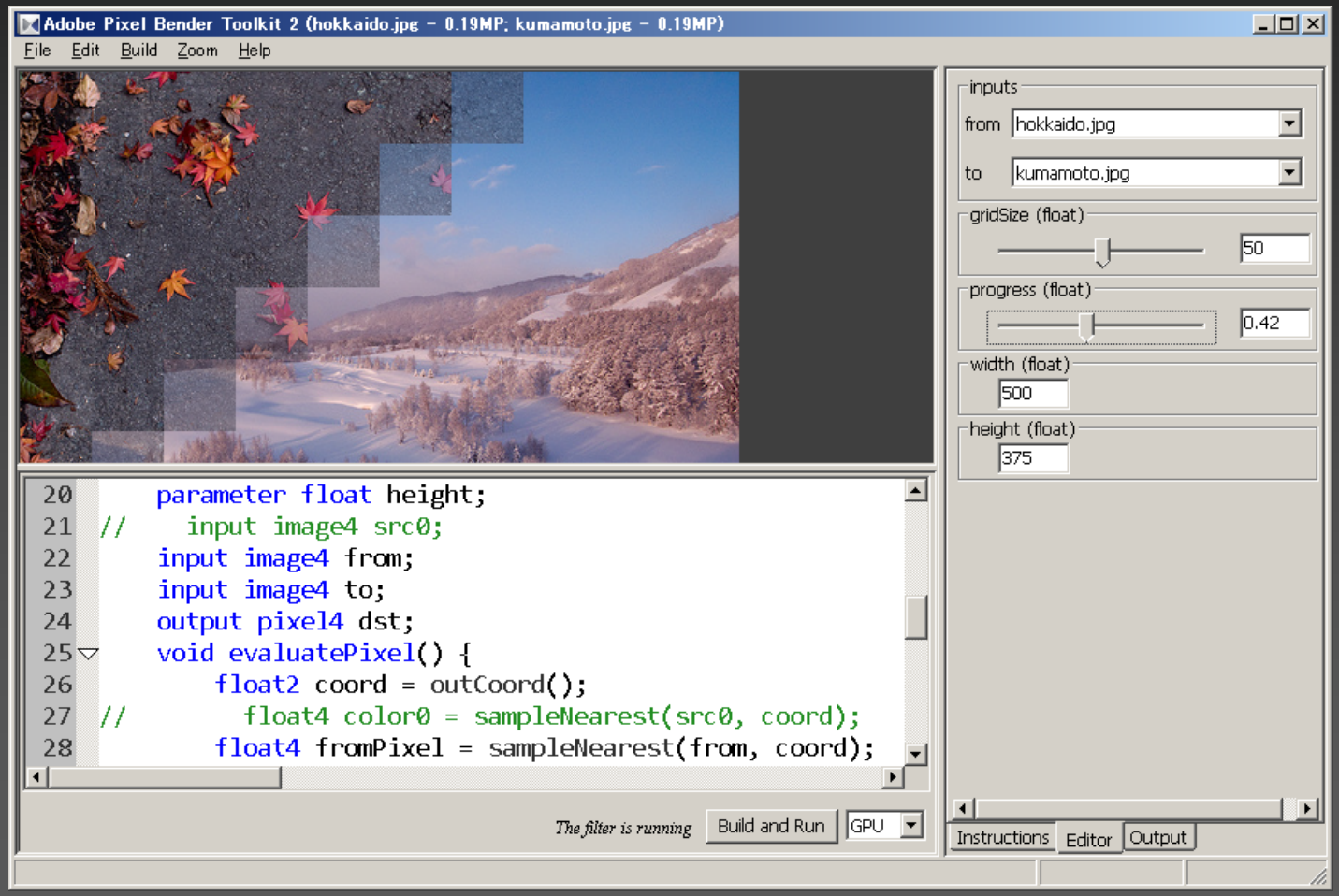
Toolkit で PBK 作成中は…

- 標準入力は不要
  (レンダリングできなくなるので削除)

```
parameter float progress;
parameter float width;
parameter float height;
// input image4 src0;
input image4 from;
input image4 to;
output pixel4 dst;
void evaluatePixel() {
//    dst = sampleNearest(src0, outCoord());
}
```

© 2010 taiga.jp

こんな感じ



© 2010 taiga.jp

AS に PBJ を埋め込む

- 埋め込み例

```
import flash.errors.IllegalOperationError;
import flash.utils.ByteArray;
import spark.effects.AnimateTransitionShader;
public class GridWipeCrossFade extends AnimateTransitionShader {
    [Embed(source="GridWipeCrossFade.pbj", mimeType="application/octet-stream")]
    private static var GridWipeCrossFadeShaderClass:Class;
    private static var gridWipeCrossFadeShaderCode:ByteArray
     = new GridWipeCrossFadeShaderClass();
    public function set gridSize(v:Number):void {
        if(v < 10 || v > 100) {
            throw new IllegalOperationError("enter a number between 10 and 100");
        }
        shaderProperties = {gridSize:v};
    }
    public function GridWipeCrossFade(target:Object=null) {
        super(target);
        shaderByteCode = gridWipeCrossFadeShaderCode;
    }
}
```

© 2010 taiga.jp

GridWipeCrossFade

© 2010 taiga.jp

考え方

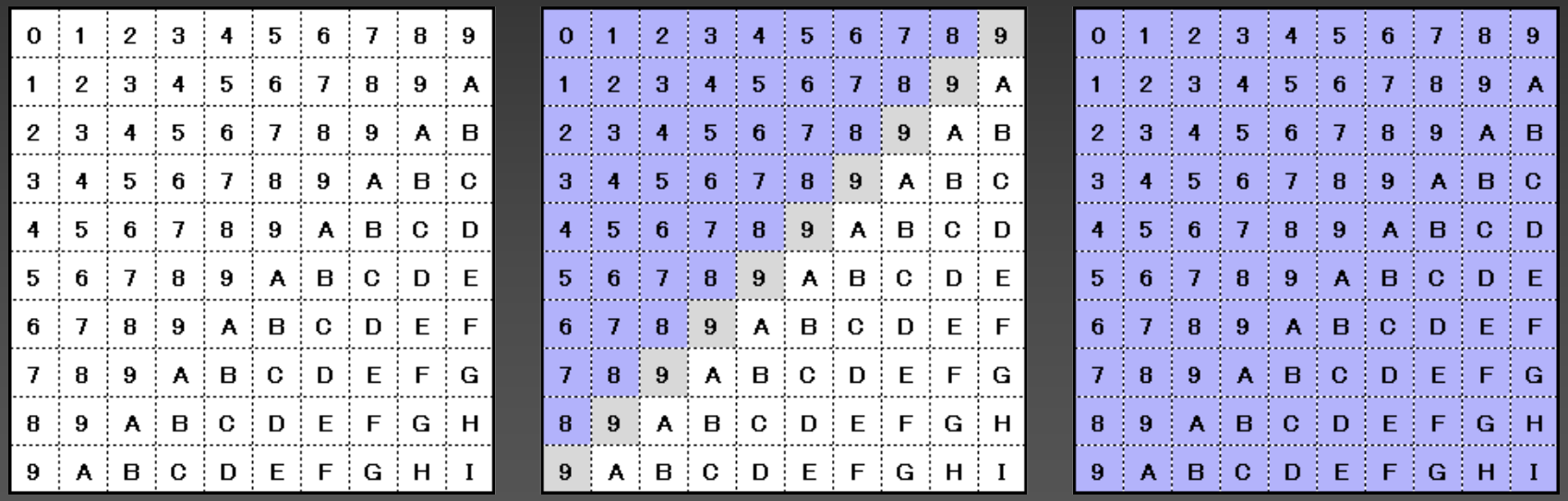
- 画素空間に階層情報を仮定する

0	1	2	3	4	5	6	7	8	9
1	2	3	4	5	6	7	8	9	A
2	3	4	5	6	7	8	9	A	B
3	4	5	6	7	8	9	A	B	C
4	5	6	7	8	9	A	B	C	D
5	6	7	8	9	A	B	C	D	E
6	7	8	9	A	B	C	D	E	F
7	8	9	A	B	C	D	E	F	G
8	9	A	B	C	D	E	F	G	H
9	A	B	C	D	E	F	G	H	I

© 2010 taiga.jp

考え方

- パラメータ (progress) の変化に合わせた推移

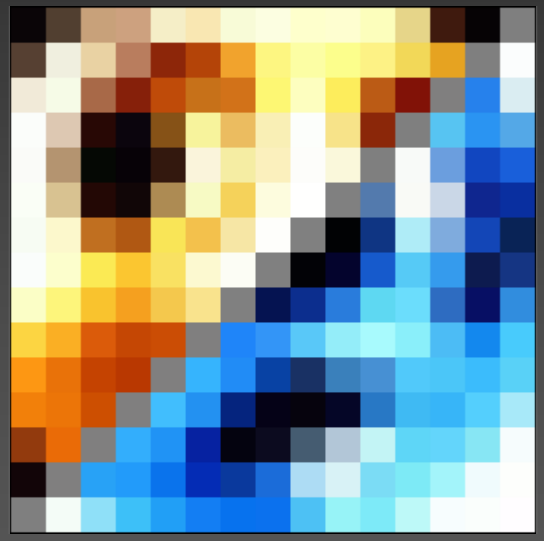


© 2010 taiga.jp

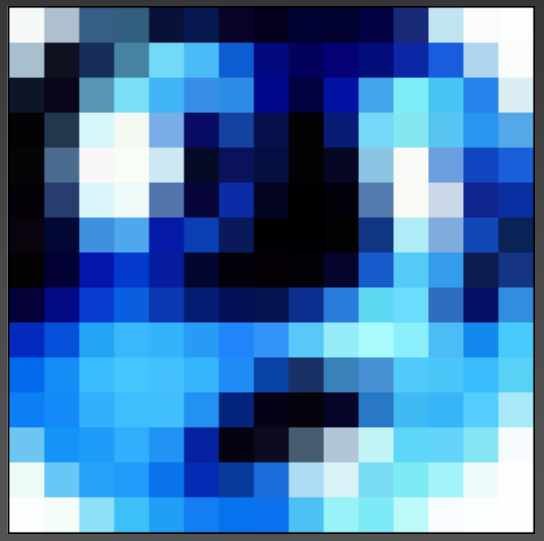
考え方

- パラメータ (progress) の変化に合わせた推移

0.0



0.5



1.0

© 2010 taiga.jp

考え方

- パラメータ (progress) から階層情報の算出

```
float hRectCount = ceil(width / gridSize);

float vRectCount = ceil(height / gridSize);

float maxLevel = hRectCount + vRectCount;

float realProg = floor(maxLevel * progress) - 1.0;
```

© 2010 taiga.jp

考え方

- サンプリングした座標から階層情報の算出

```
float2 coord = outCoord();

float px = floor(coord.x / gridSize);

float py = floor(coord.y / gridSize);

float currentLevel = px + py;
```

© 2010 taiga.jp

考え方

- 算出した階層情報を比較して出力するピクセル情報を定義

```
if (realProg < currentLevel)
    dst = fromPixel;

else if (realProg > currentLevel)
    dst = toPixel;

else if (realProg == currentLevel)
    dst = mix(fromPixel, toPixel, 0.5);
```

© 2010 taiga.jp

実に…

おしゃれ でシンプル

© 2010 taiga.jp

参考サイト

- BeInteractive! - Pixel Bender と戯れる
  http://www.be-interactive.org/works/20081126/be-lt04-pixelbender.pdf
- CrossFade.as
  http://opensource.adobe.com/svn/opensource/flex/sdk/trunk/frameworks/projects/spark/src/spark/effects/CrossFade.as
- CrossFade.pbk
  http://opensource.adobe.com/svn/opensource/flex/sdk/trunk/frameworks/projects/spark/src/spark/effects/CrossFade.pbk

© 2010 taiga.jp

参考書籍

詳解 画像処理プログラミング
C言語で実装する画像処理アルゴリズムのすべて

出版社: ソフトバンククリエイティブ
ISBN-10: 4797344377
ISBN-13: 978-4797344370

© 2010 taiga.jp