CODE 1: C code

```c
//by baowenbo
/* hello world*/

#include"stdio.h"
int main(int argc,char * argv){
    printf("output a value\n");
}
```

CODE 2: Some Code

```
    public void here() {
        goes().the().code()
    }
```

CODE 3: A sample

```
public void here() {
    goes().the().code()

}
```

CODE 4: A sample

```
#include "YUV.SJTU_headers.h"
//#include "upscale.h"

#define HF          (8.0f)
#define a           (-0.50000000f)
__inline__ __device__ float cubic(float x){
    float abs_x = fabsf(x);

    if (abs_x <= 1.0f){
        return (a + 2)*abs_x *abs_x *abs_x - (a + 3)*abs_x *abs_x + 1;
    }
    else if (abs_x < 2.0f){
        return a * abs_x *abs_x *abs_x - 5 * a *abs_x *abs_x + 8 * a *
            abs_x - 4 * a;
    }
    else {
        return 0.0f;
    }
}
__inline__ __device__ float linear(float x){
    return x;
}


__global__ void DownSample(unsigned char * In, int iw, int ih,
    float * downbicu, int dw, int dh,
    float downscale_x, float downscale_y
    ) {
    const int dstBaseX = blockIdx.x * blockDim.x + threadIdx.x;
```

```
    const int dstBaseY = blockIdx.y * blockDim.y + threadIdx.y;

    float downbicu_value = 0.0f;
    //float downbili_value = 0.0f;

    float srcY, srcX;
    int srcY0, srcX0; int srcY1, srcX1; int srcY2, srcX2; int srcY3,
        srcX3;


    float cubic_X[4], cubic_Y[4];
    //float linear_X[2], linear_Y[2];

    srcX = (dstBaseX + 1) / downscale_x + 0.5f*(1.0f - 1.0f /
        downscale_x) - 1;
    srcY = (dstBaseY + 1) / downscale_y + 0.5f*(1.0f - 1.0f /
        downscale_y) - 1;

    srcX0 = floorf(srcX) - 1;
    srcY0 = floorf(srcY) - 1;
    srcX1 = srcX0 + 1;
    srcY1 = srcY0 + 1;
    srcX2 = srcX0 + 2;
    srcY2 = srcY0 + 2;
    srcX3 = srcX0 + 3;
    srcY3 = srcY0 + 3;

    cubic_X[0] = cubic(srcX - srcX0);
    cubic_X[1] = cubic(srcX - srcX1);
    cubic_X[2] = cubic(srcX - srcX2);
    cubic_X[3] = cubic(srcX - srcX3);

    cubic_Y[0] = cubic(srcY - srcY0);
    cubic_Y[1] = cubic(srcY - srcY1);
    cubic_Y[2] = cubic(srcY - srcY2);
    cubic_Y[3] = cubic(srcY - srcY3);

    //linear_X[0] = linear(srcX2 - srcX);
    //linear_X[1] = linear(srcX - srcX1);

    //linear_Y[0] = linear(srcY2 - srcY);
    //linear_Y[1] = linear(srcY - srcY1);


    srcX0 = min(max(srcX0, 0), iw - 1);// (srcX0 < 0) ? 0 : srcX0;
    srcY0 = min(max(srcY0, 0), ih - 1);// (srcY0 < 0) ? 0 : srcY0;
    srcX1 = min(max(srcX1, 0), iw - 1);// (srcX1 < 0) ? 0 : srcX1;
    srcY1 = min(max(srcY1, 0), ih - 1);//(srcY1 < 0) ? 0 : srcY1;
    srcX2 = min(max(srcX2, 0), iw - 1);// (srcX2 > validW - 1) ?
        validW - 1 : srcX2;
    srcY2 = min(max(srcY2, 0), ih - 1);//(srcY2 > validH - 1) ?
        validH - 1 : srcY2;
    srcX3 = min(max(srcX3, 0), iw - 1);// (srcX3 > validW - 1) ?
        validW - 1 : srcX3;
    srcY3 = min(max(srcY3, 0), ih - 1);//(srcY3 > validH - 1) ?
        validH - 1 : srcY3;

```

```
    downbicu_value =
       cubic_X[0] * cubic_Y[0] * In[srcY0 * iw + srcX0] +
       cubic_X[0] * cubic_Y[1] * In[srcY1 * iw + srcX0] +
       cubic_X[0] * cubic_Y[2] * In[srcY2 * iw + srcX0] +
       cubic_X[0] * cubic_Y[3] * In[srcY3 * iw + srcX0] +
       cubic_X[1] * cubic_Y[0] * In[srcY0 * iw + srcX1] +
       cubic_X[1] * cubic_Y[1] * In[srcY1 * iw + srcX1] +
       cubic_X[1] * cubic_Y[2] * In[srcY2 * iw + srcX1] +
       cubic_X[1] * cubic_Y[3] * In[srcY3 * iw + srcX1] +
       cubic_X[2] * cubic_Y[0] * In[srcY0 * iw + srcX2] +
       cubic_X[2] * cubic_Y[1] * In[srcY1 * iw + srcX2] +
       cubic_X[2] * cubic_Y[2] * In[srcY2 * iw + srcX2] +
       cubic_X[2] * cubic_Y[3] * In[srcY3 * iw + srcX2] +
       cubic_X[3] * cubic_Y[0] * In[srcY0 * iw + srcX3] +
       cubic_X[3] * cubic_Y[1] * In[srcY1 * iw + srcX3] +
       cubic_X[3] * cubic_Y[2] * In[srcY2 * iw + srcX3] +
       cubic_X[3] * cubic_Y[3] * In[srcY3 * iw + srcX3];

    //downbili_value =
    // linear_X[0] * linear_Y[0] * In[srcY1 * iw + srcX1] +
    // linear_X[0] * linear_Y[1] * In[srcY2 * iw + srcX1] +
    // linear_X[1] * linear_Y[0] * In[srcY1 * iw + srcX2] +
    // linear_X[1] * linear_Y[1] * In[srcY2 * iw + srcX2];

    if (dstBaseY < dh && dstBaseX < dw){
       downbicu[dstBaseY * dw + dstBaseX] = downbicu_value;
       //downbili[dstBaseY * dw + dstBaseX] = downbili_value;
    }
}

__global__ void UpEnhanceLL(
    float * downbicu, int dw, int dh,
    unsigned char *In, unsigned char * Out, int ow, int oh,
    float upscale_x, float upscale_y
    ){
    const int dstBaseX = blockIdx.x * blockDim.x + threadIdx.x;
    const int dstBaseY = blockIdx.y * blockDim.y + threadIdx.y;

    float upbicu_value = 0.0f;
    //float upbili_value = 0.0f;


    float srcY, srcX;

    int srcY0, srcX0; int srcY1, srcX1; int srcY2, srcX2; int srcY3,
        srcX3;

    float cubic_X[4], cubic_Y[4];
    //float linear_X[2], linear_Y[2];

    if (dstBaseX < ow / 2 - 1){
    // i just use the local coordinate instead of global coordinate
        ....this may cause some block effect ....let's see
    srcX = (dstBaseX + 1) / upscale_x + 0.5f*(1.0f - 1.0f / upscale_x
        ) - 1;
    srcY = (dstBaseY + 1) / upscale_y + 0.5f*(1.0f - 1.0f / upscale_y
        ) - 1;
```

```
132
      srcX0 = floorf(srcX) - 1;
134   srcY0 = floorf(srcY) - 1;
      srcX1 = srcX0 + 1;
136   srcY1 = srcY0 + 1;
      srcX2 = srcX0 + 2;
138   srcY2 = srcY0 + 2;
      srcX3 = srcX0 + 3;
140   srcY3 = srcY0 + 3;

142   cubic_X[0] = cubic(srcX - srcX0);
      cubic_X[1] = cubic(srcX - srcX1);
144   cubic_X[2] = cubic(srcX - srcX2);
      cubic_X[3] = cubic(srcX - srcX3);
146
      cubic_Y[0] = cubic(srcY - srcY0);
148   cubic_Y[1] = cubic(srcY - srcY1);
      cubic_Y[2] = cubic(srcY - srcY2);
150   cubic_Y[3] = cubic(srcY - srcY3);

152   //linear_X[0] = linear(srcX2 - srcX);
      //linear_X[1] = linear(srcX - srcX1);
154
      //linear_Y[0] = linear(srcY2 - srcY);
156   //linear_Y[1] = linear(srcY - srcY1);

158   srcX0 = min(max(srcX0, 0), dw - 1);// (srcX0 < 0) ? 0 : srcX0;
      srcY0 = min(max(srcY0, 0), dh - 1);// (srcY0 < 0) ? 0 : srcY0;
160   srcX1 = min(max(srcX1, 0), dw - 1);// (srcX1 < 0) ? 0 : srcX1;
      srcY1 = min(max(srcY1, 0), dh - 1);//(srcY1 < 0) ? 0 : srcY1;
162   srcX2 = min(max(srcX2, 0), dw - 1);// (srcX2 > validW - 1) ?
          validW - 1 : srcX2;
      srcY2 = min(max(srcY2, 0), dh - 1);//(srcY2 > validH - 1) ?
          validH - 1 : srcY2;
164   srcX3 = min(max(srcX3, 0), dw - 1);// (srcX3 > validW - 1) ?
          validW - 1 : srcX3;
      srcY3 = min(max(srcY3, 0), dh - 1);//(srcY3 > validH - 1) ?
          validH - 1 : srcY3;
166

168   upbicu_value =
         cubic_X[0] * cubic_Y[0] * downbicu[srcY0 * dw + srcX0] +
170      cubic_X[0] * cubic_Y[1] * downbicu[srcY1 * dw + srcX0] +
         cubic_X[0] * cubic_Y[2] * downbicu[srcY2 * dw + srcX0] +
172      cubic_X[0] * cubic_Y[3] * downbicu[srcY3 * dw + srcX0] +
         cubic_X[1] * cubic_Y[0] * downbicu[srcY0 * dw + srcX1] +
174      cubic_X[1] * cubic_Y[1] * downbicu[srcY1 * dw + srcX1] +
         cubic_X[1] * cubic_Y[2] * downbicu[srcY2 * dw + srcX1] +
176      cubic_X[1] * cubic_Y[3] * downbicu[srcY3 * dw + srcX1] +
         cubic_X[2] * cubic_Y[0] * downbicu[srcY0 * dw + srcX2] +
178      cubic_X[2] * cubic_Y[1] * downbicu[srcY1 * dw + srcX2] +
         cubic_X[2] * cubic_Y[2] * downbicu[srcY2 * dw + srcX2] +
180      cubic_X[2] * cubic_Y[3] * downbicu[srcY3 * dw + srcX2] +
         cubic_X[3] * cubic_Y[0] * downbicu[srcY0 * dw + srcX3] +
182      cubic_X[3] * cubic_Y[1] * downbicu[srcY1 * dw + srcX3] +
         cubic_X[3] * cubic_Y[2] * downbicu[srcY2 * dw + srcX3] +
184      cubic_X[3] * cubic_Y[3] * downbicu[srcY3 * dw + srcX3];
```

```
186     //upbili_value =
        // linear_X[0] * linear_Y[0] * downbili[srcY1 * dw + srcX1] +
188     // linear_X[0] * linear_Y[1] * downbili[srcY2 * dw + srcX1] +
        // linear_X[1] * linear_Y[0] * downbili[srcY1 * dw + srcX2] +
190     // linear_X[1] * linear_Y[1] * downbili[srcY2 * dw + srcX2];

192
        upbicu_value = In[dstBaseY * ow + dstBaseX] + HF*(In[dstBaseY *
            ow + dstBaseX] - upbicu_value);
194     }
        else if (dstBaseX == ow / 2 - 1 || dstBaseX == ow / 2){
196        upbicu_value = 255;
        }
198     else{
           upbicu_value = In[dstBaseY * ow + dstBaseX - ow / 2];
200     }

202     if (dstBaseY < oh && dstBaseX < ow){
           Out[dstBaseY * ow + dstBaseX] = min(max(unsigned int(
               upbicu_value), 0), 255);
204     }
    }
206
    void OptBicu_BicuexeckernelLL(unsigned char * In, int iw, int ih,
208     float * downbicu, int dw, int dh, float downscale_x, float
            downscale_y,
        unsigned char * Out, int ow, int oh, float upscale_x, float
            upscale_y
210     ){

212     static dim3 grid;
        static dim3 block;
214
        block.x = 32;
216     block.y = 32;
        block.z = 1;
218     grid.x = (dw + block.x - 1) / block.x;
        grid.y = (dh + block.y - 1) / block.y;
220     grid.z = 1;

222     DownSample << <grid, block >> >(In, iw, ih, downbicu, dw, dh,
            downscale_x, downscale_y);

224     grid.x = (ow + block.x - 1) / block.x;
        grid.y = (oh + block.y - 1) / block.y;
226     grid.z = 1;


228
        //Sheme Four:
230        //left side ==> I + 4 *(I - upbicu(downbicu(I)))
           //right side ==> I
232     UpEnhanceLL << <grid, block >> >(downbicu, dw, dh,In, Out, ow, oh
            , upscale_x, upscale_y);


234
    }
```

```
236
238
__global__ void UpEnhanceLR(
240     float * downbicu, int dw, int dh,
        unsigned char *In, unsigned char * Out, int ow, int oh,
242     float upscale_x, float upscale_y
        ){
244     const int dstBaseX = blockIdx.x * blockDim.x + threadIdx.x;
        const int dstBaseY = blockIdx.y * blockDim.y + threadIdx.y;
246
        float upbicu_value = 0.0f;
248     //float upbili_value = 0.0f;
250
        float srcY, srcX;
252
        int srcY0, srcX0; int srcY1, srcX1; int srcY2, srcX2; int srcY3,
            srcX3;
254
        float cubic_X[4], cubic_Y[4];
256     //float linear_X[2], linear_Y[2];
258
        // i just use the local coordinate instead of global coordinate
            ....this may cause some block effect ....let's see
260     srcX = (dstBaseX + 1) / upscale_x + 0.5f*(1.0f - 1.0f / upscale_x
            ) - 1;
        srcY = (dstBaseY + 1) / upscale_y + 0.5f*(1.0f - 1.0f / upscale_y
            ) - 1;
262
        srcX0 = floorf(srcX) - 1;
264     srcY0 = floorf(srcY) - 1;
        srcX1 = srcX0 + 1;
266     srcY1 = srcY0 + 1;
        srcX2 = srcX0 + 2;
268     srcY2 = srcY0 + 2;
        srcX3 = srcX0 + 3;
270     srcY3 = srcY0 + 3;

272     cubic_X[0] = cubic(srcX - srcX0);
        cubic_X[1] = cubic(srcX - srcX1);
274     cubic_X[2] = cubic(srcX - srcX2);
        cubic_X[3] = cubic(srcX - srcX3);
276
        cubic_Y[0] = cubic(srcY - srcY0);
278     cubic_Y[1] = cubic(srcY - srcY1);
        cubic_Y[2] = cubic(srcY - srcY2);
280     cubic_Y[3] = cubic(srcY - srcY3);

282     //linear_X[0] = linear(srcX2 - srcX);
        //linear_X[1] = linear(srcX - srcX1);
284
        //linear_Y[0] = linear(srcY2 - srcY);
286     //linear_Y[1] = linear(srcY - srcY1);

288     srcX0 = min(max(srcX0, 0), dw - 1);// (srcX0 < 0) ? 0 : srcX0;
```

```
        srcY0 = min(max(srcY0, 0), dh - 1);// (srcY0 < 0) ? 0 : srcY0;
290     srcX1 = min(max(srcX1, 0), dw - 1);// (srcX1 < 0) ? 0 : srcX1;
        srcY1 = min(max(srcY1, 0), dh - 1);//(srcY1 < 0) ? 0 : srcY1;
292     srcX2 = min(max(srcX2, 0), dw - 1);// (srcX2 > validW - 1) ?
            validW - 1 : srcX2;
        srcY2 = min(max(srcY2, 0), dh - 1);//(srcY2 > validH - 1) ?
            validH - 1 : srcY2;
294     srcX3 = min(max(srcX3, 0), dw - 1);// (srcX3 > validW - 1) ?
            validW - 1 : srcX3;
        srcY3 = min(max(srcY3, 0), dh - 1);//(srcY3 > validH - 1) ?
            validH - 1 : srcY3;
296
298     upbicu_value =
           cubic_X[0] * cubic_Y[0] * downbicu[srcY0 * dw + srcX0] +
300        cubic_X[0] * cubic_Y[1] * downbicu[srcY1 * dw + srcX0] +
           cubic_X[0] * cubic_Y[2] * downbicu[srcY2 * dw + srcX0] +
302        cubic_X[0] * cubic_Y[3] * downbicu[srcY3 * dw + srcX0] +
           cubic_X[1] * cubic_Y[0] * downbicu[srcY0 * dw + srcX1] +
304        cubic_X[1] * cubic_Y[1] * downbicu[srcY1 * dw + srcX1] +
           cubic_X[1] * cubic_Y[2] * downbicu[srcY2 * dw + srcX1] +
306        cubic_X[1] * cubic_Y[3] * downbicu[srcY3 * dw + srcX1] +
           cubic_X[2] * cubic_Y[0] * downbicu[srcY0 * dw + srcX2] +
308        cubic_X[2] * cubic_Y[1] * downbicu[srcY1 * dw + srcX2] +
           cubic_X[2] * cubic_Y[2] * downbicu[srcY2 * dw + srcX2] +
310        cubic_X[2] * cubic_Y[3] * downbicu[srcY3 * dw + srcX2] +
           cubic_X[3] * cubic_Y[0] * downbicu[srcY0 * dw + srcX3] +
312        cubic_X[3] * cubic_Y[1] * downbicu[srcY1 * dw + srcX3] +
           cubic_X[3] * cubic_Y[2] * downbicu[srcY2 * dw + srcX3] +
314        cubic_X[3] * cubic_Y[3] * downbicu[srcY3 * dw + srcX3];

316     //upbili_value =
        // linear_X[0] * linear_Y[0] * downbili[srcY1 * dw + srcX1] +
318     // linear_X[0] * linear_Y[1] * downbili[srcY2 * dw + srcX1] +
        // linear_X[1] * linear_Y[0] * downbili[srcY1 * dw + srcX2] +
320     // linear_X[1] * linear_Y[1] * downbili[srcY2 * dw + srcX2];

322     if (dstBaseX < ow / 2 - 1){
           upbicu_value = In[dstBaseY * ow + dstBaseX] + HF*(In[dstBaseY
               * ow + dstBaseX] - upbicu_value);
324     }
        else if (dstBaseX == ow / 2 - 1 || dstBaseX == ow / 2){
326        upbicu_value = 255;
        }
328     else{
           upbicu_value = In[dstBaseY * ow + dstBaseX];
330     }

332     if (dstBaseY < oh && dstBaseX < ow){
           Out[dstBaseY * ow + dstBaseX] = min(max(unsigned int(
               upbicu_value), 0), 255);
334     }
    }
336

338 void OptBicu_BicuexeckernelLR(unsigned char * In, int iw, int ih,
        float * downbicu, int dw, int dh, float downscale_x, float
```

```
        downscale_y,
    unsigned char * Out, int ow, int oh, float upscale_x, float
        upscale_y
    ){

    static dim3 grid;
    static dim3 block;

    block.x = 32;
    block.y = 32;
    block.z = 1;
    grid.x = (dw + block.x - 1) / block.x;
    grid.y = (dh + block.y - 1) / block.y;
    grid.z = 1;

    DownSample << <grid, block >> >(In, iw, ih, downbicu, dw, dh,
        downscale_x, downscale_y);

    grid.x = (ow + block.x - 1) / block.x;
    grid.y = (oh + block.y - 1) / block.y;
    grid.z = 1;


    //Sheme Four:
    //left side ==> I + 4 *(I - upbicu(downbicu(I)))
    //right side ==> I
    UpEnhanceLR << <grid, block >> >(downbicu, dw, dh, In, Out, ow,
        oh, upscale_x, upscale_y);


}

__global__ void UpEnhance(
    float * downbicu, int dw, int dh,
    unsigned char *In, unsigned char * Out, int ow, int oh,
    float upscale_x, float upscale_y
    ){
    const int dstBaseX = blockIdx.x * blockDim.x + threadIdx.x;
    const int dstBaseY = blockIdx.y * blockDim.y + threadIdx.y;

    float upbicu_value = 0.0f;
    //float upbili_value = 0.0f;


    float srcY, srcX;

    int srcY0, srcX0; int srcY1, srcX1; int srcY2, srcX2; int srcY3,
        srcX3;

    float cubic_X[4], cubic_Y[4];
    //float linear_X[2], linear_Y[2];


    // i just use the local coordinate instead of global coordinate
        ....this may cause some block effect ....let's see
    srcX = (dstBaseX + 1) / upscale_x + 0.5f*(1.0f - 1.0f / upscale_x
        ) - 1;
```

```
390     srcY = (dstBaseY + 1) / upscale_y + 0.5f*(1.0f - 1.0f / upscale_y
            ) - 1;

392     srcX0 = floorf(srcX) - 1;
        srcY0 = floorf(srcY) - 1;
394     srcX1 = srcX0 + 1;
        srcY1 = srcY0 + 1;
396     srcX2 = srcX0 + 2;
        srcY2 = srcY0 + 2;
398     srcX3 = srcX0 + 3;
        srcY3 = srcY0 + 3;
400
        cubic_X[0] = cubic(srcX - srcX0);
402     cubic_X[1] = cubic(srcX - srcX1);
        cubic_X[2] = cubic(srcX - srcX2);
404     cubic_X[3] = cubic(srcX - srcX3);

406     cubic_Y[0] = cubic(srcY - srcY0);
        cubic_Y[1] = cubic(srcY - srcY1);
408     cubic_Y[2] = cubic(srcY - srcY2);
        cubic_Y[3] = cubic(srcY - srcY3);
410
        //linear_X[0] = linear(srcX2 - srcX);
412     //linear_X[1] = linear(srcX - srcX1);

414     //linear_Y[0] = linear(srcY2 - srcY);
        //linear_Y[1] = linear(srcY - srcY1);
416
        srcX0 = min(max(srcX0, 0), dw - 1);// (srcX0 < 0) ? 0 : srcX0;
418     srcY0 = min(max(srcY0, 0), dh - 1);// (srcY0 < 0) ? 0 : srcY0;
        srcX1 = min(max(srcX1, 0), dw - 1);// (srcX1 < 0) ? 0 : srcX1;
420     srcY1 = min(max(srcY1, 0), dh - 1);//(srcY1 < 0) ? 0 : srcY1;
        srcX2 = min(max(srcX2, 0), dw - 1);// (srcX2 > validW - 1) ?
            validW - 1 : srcX2;
422     srcY2 = min(max(srcY2, 0), dh - 1);//(srcY2 > validH - 1) ?
            validH - 1 : srcY2;
        srcX3 = min(max(srcX3, 0), dw - 1);// (srcX3 > validW - 1) ?
            validW - 1 : srcX3;
424     srcY3 = min(max(srcY3, 0), dh - 1);//(srcY3 > validH - 1) ?
            validH - 1 : srcY3;


426
        upbicu_value =
428        cubic_X[0] * cubic_Y[0] * downbicu[srcY0 * dw + srcX0] +
           cubic_X[0] * cubic_Y[1] * downbicu[srcY1 * dw + srcX0] +
430        cubic_X[0] * cubic_Y[2] * downbicu[srcY2 * dw + srcX0] +
           cubic_X[0] * cubic_Y[3] * downbicu[srcY3 * dw + srcX0] +
432        cubic_X[1] * cubic_Y[0] * downbicu[srcY0 * dw + srcX1] +
           cubic_X[1] * cubic_Y[1] * downbicu[srcY1 * dw + srcX1] +
434        cubic_X[1] * cubic_Y[2] * downbicu[srcY2 * dw + srcX1] +
           cubic_X[1] * cubic_Y[3] * downbicu[srcY3 * dw + srcX1] +
436        cubic_X[2] * cubic_Y[0] * downbicu[srcY0 * dw + srcX2] +
           cubic_X[2] * cubic_Y[1] * downbicu[srcY1 * dw + srcX2] +
438        cubic_X[2] * cubic_Y[2] * downbicu[srcY2 * dw + srcX2] +
           cubic_X[2] * cubic_Y[3] * downbicu[srcY3 * dw + srcX2] +
440        cubic_X[3] * cubic_Y[0] * downbicu[srcY0 * dw + srcX3] +
           cubic_X[3] * cubic_Y[1] * downbicu[srcY1 * dw + srcX3] +
```

```
442         cubic_X[3] * cubic_Y[2] * downbicu[srcY2 * dw + srcX3] +
            cubic_X[3] * cubic_Y[3] * downbicu[srcY3 * dw + srcX3];
444
        //upbili_value =
446     // linear_X[0] * linear_Y[0] * downbili[srcY1 * dw + srcX1] +
        // linear_X[0] * linear_Y[1] * downbili[srcY2 * dw + srcX1] +
448     // linear_X[1] * linear_Y[0] * downbili[srcY1 * dw + srcX2] +
        // linear_X[1] * linear_Y[1] * downbili[srcY2 * dw + srcX2];
450
        upbicu_value = In[dstBaseY * ow + dstBaseX] + HF*(In[dstBaseY *
            ow + dstBaseX] - upbicu_value);
452
        if (dstBaseY < oh && dstBaseX < ow){
454        Out[dstBaseY * ow + dstBaseX] = min(max(unsigned int(
               upbicu_value), 0), 255);
        }
456 }


458
    void OptBicu_Bicuexeckernel(unsigned char * In, int iw, int ih,
460     float * downbicu, int dw, int dh, float downscale_x, float
            downscale_y,
        unsigned char * Out, int ow, int oh, float upscale_x, float
            upscale_y
462     ){

464     static dim3 grid;
        static dim3 block;
466
        block.x = 32;
468     block.y = 32;
        block.z = 1;
470     grid.x = (dw + block.x - 1) / block.x;
        grid.y = (dh + block.y - 1) / block.y;
472     grid.z = 1;

474     DownSample << <grid, block >> >(In, iw, ih, downbicu, dw, dh,
            downscale_x, downscale_y);

476     grid.x = (ow + block.x - 1) / block.x;
        grid.y = (oh + block.y - 1) / block.y;
478     grid.z = 1;


480
        //Sheme Four:
482     //left side ==> I + 4 *(I - upbicu(downbicu(I)))
        //right side ==> I
484     UpEnhance << <grid, block >> >(downbicu, dw, dh, In, Out, ow, oh,
             upscale_x, upscale_y);
        printf("Here␣I␣anm\n\year");
486
    }
```