Parallel Algorithms Homework № 3

Aditya Bhamidipati, Georgia State University 04/18/2020

* Source code snippets displayed are for understanding purpose only and hence are logic centered. For full source code refer the attached .cu files

Problem 1

(50 pts) Write a CUDA program for calculating the prefix sums of an arbitrary-length array as described in Section 8.6 of the Kirk and Hwu book. You should follow their code snippets and input/output conventions.

Listing 1: Source Code – Parallel Prefix Sum method.

```
1
2  void psum(const unsigned int * const h_in, unsigned int * const h_out, ↩
       const size_t len)
3  {
4    int nblocks = n/block_size;
5
6    // allocate memories on gpu
7    unsigned int *d_scan, *d_sums, *d_incr;
8
9
10   // scan array by blocks (block_size = 2 * num threads)
11   block_psum<<<nblocks, nthreads, smem>>>(d_in, d_scan, d_sums, block_size↩
         );
12
13   // scan block sums
14   // TODO case when nblocks is bigger than block_size (see TODO 1)
15   block_psum<<<1, nthreads, smem>>>(d_sums, d_incr, NULL, block_size);
16
17   // scatter block sums back to scanned blocks
18   scatter_incr<<<nblocks, nthreads>>>(d_scan, d_incr);
19
20 }
```

Following Listing 1... Here we initialize our GPU memory to hold our array elements. Scan all the array elements by blocks. Then we perform sum among the blocks separately. The idea is to scatter the block of arrays among GPUs and calculate the prefix-sums in a parallel fashion. Then we scatter the blocks back to their original places so as to give us the original array structure and every index position holds sum till that position.

Listing 2: Source Code – Parallel Block Prefix Sum method.

```
1
2  __global__
3  void block_psum(const unsigned int * const g_in, unsigned int * const ↩
       g_out, unsigned int * const g_sums, const size_t n)
4  {
5
6    // up sweep
7
8    for (int d = n >> 1; d > 0; d >>= 1)
9    {
10     __syncthreads();
11
12     if (tx < d)
13     {
14       int ai = offset * (2*tx+1) - 1;
15       int bi = offset * (2*tx+2) - 1;
16
17       smem[bi] += smem[ai];
18     }
19     offset <<= 1;
20   }
21
22   // save block sum and clear last element
23   if (tx == 0) {
24     if (g_sums != NULL)
25       g_sums[blockIdx.x] = smem[n-1];
26     smem[n-1] = 0;
27   }
28
29   // down sweep
30
31   for (int d = 1; d < n; d <<= 1)
32   {
33     offset >>= 1;
34     __syncthreads();
35
36     if (tx < d)
37     {
38       int ai = offset * (2*tx+1) - 1;
39       int bi = offset * (2*tx+2) - 1;
40
41       // swap
42       unsigned int t = smem[ai];
43       smem[ai]  = smem[bi];
44       smem[bi] += t;
45     }
```

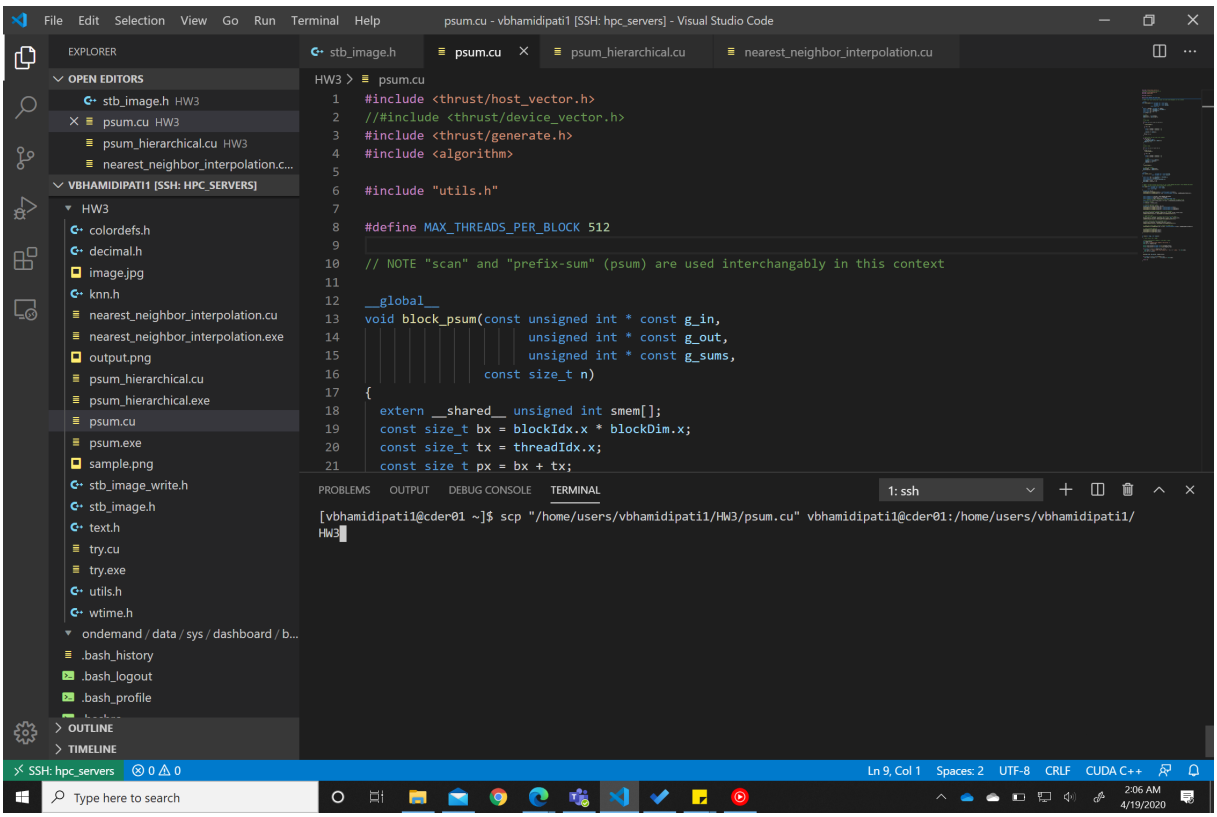
```
46    }
47    __syncthreads();
48
49    // save scan result
50    g_out[2*px]   = smem[2*tx];
51    g_out[2*px+1] = smem[2*tx+1];
52  }
```

Following Listing 2... **Upsweep:** This is also know as a reduce operation. The reduction happens in a heirarchical fashion. During the first iteration, the neighbors with one hop away are added. In the subsequent iterations neighbors with 2, 4, 8 etc., hops are added respectively until we are left with one number that is the sum of all elements of the array. **Downsweep:** We start by resetting the right most value by the identity operator. The downsweep creates a mirror image of hierarchy formed by upsweep. If a value is missing for a downsweep operation, it is simply copied from the upsweep operation.

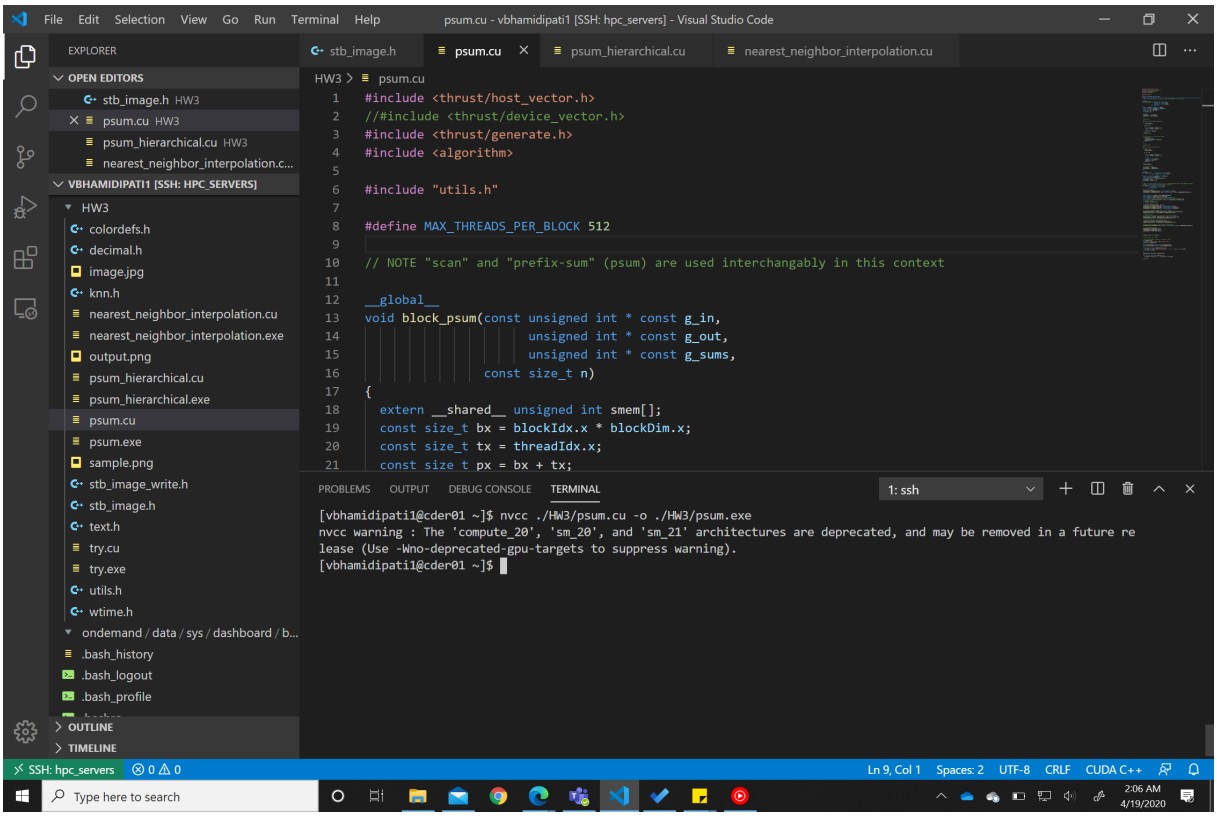
Listing 3: Source Code – Combining the scattered blocks.

```
1
2   __global__
3   void scatter_incr(      unsigned int * const d_array,
4                     const unsigned int * const d_incr)
5   {
6     const size_t bx = 2 * blockDim.x * blockIdx.x;
7     const size_t tx = threadIdx.x;
8     const unsigned int u = d_incr[blockIdx.x];
9     d_array[bx + 2*tx]   += u;
10    d_array[bx + 2*tx+1] += u;
11  }
```

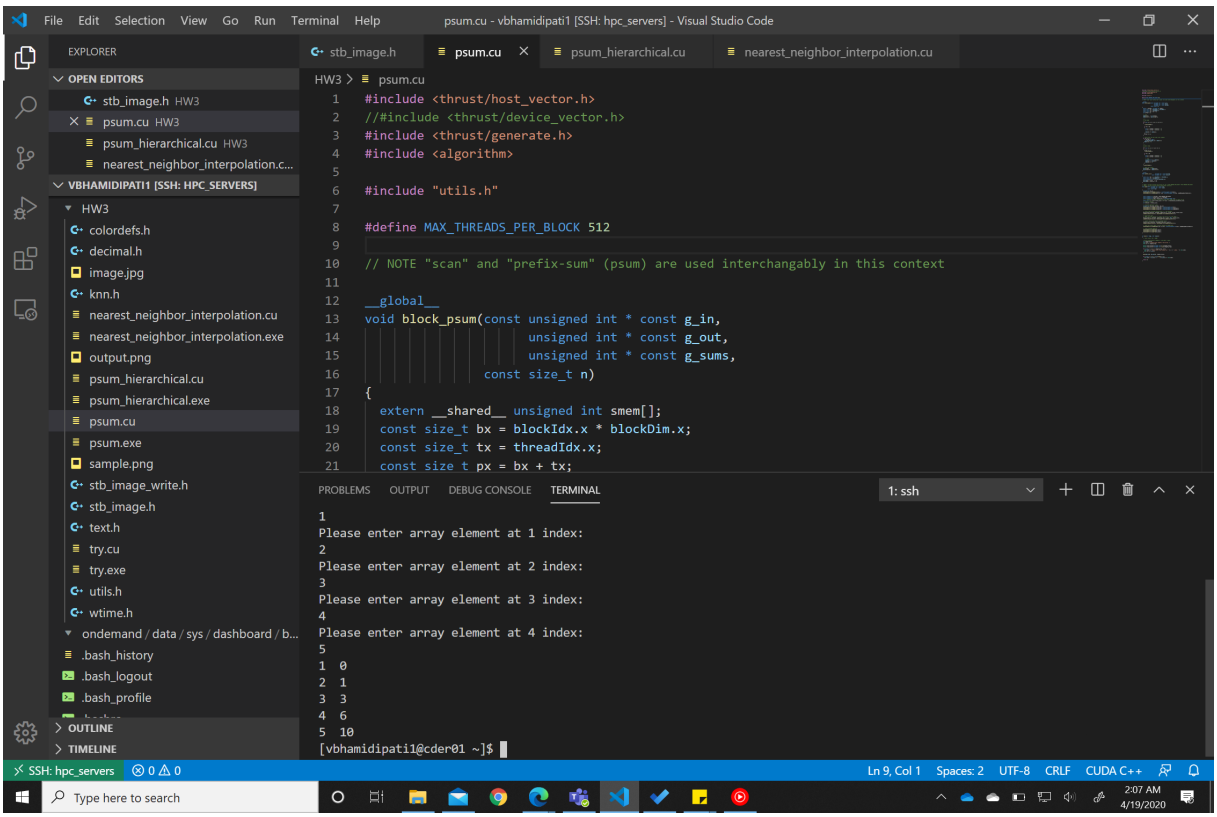
Following Listing 3... Here we combine all the scattered blocks into single array, where the value at any index gives the prefix sum till that position.


File Edit Selection View Go Run Terminal Help
psum.cu - vbhamidipati1 [SSH: hpc_servers] - Visual Studio Code
EXPLORER
OPEN EDITORS
stb_image.h HW3
psum.cu HW3
psum_hierarchical.cu HW3
nearest_neighbor_interpolation.c...
VBHAMIDIPATI1 [SSH: HPC_SERVERS]
HW3
colordefs.h
decimal.h
image.jpg
knn.h
nearest_neighbor_interpolation.cu
nearest_neighbor_interpolation.exe
output.png
psum_hierarchical.cu
psum_hierarchical.exe
psum.cu
psum.exe
sample.png
stb_image_write.h
stb_image.h
text.h
try.cu
try.exe
utils.h
wtime.h
ondemand / data / sys / dashboard / b...
.bash_history
.bash_logout
.bash_profile
OUTLINE
TIMELINE
stb_image.h
psum.cu
psum_hierarchical.cu
nearest_neighbor_interpolation.cu
HW3 > psum.cu
#include <thrust/host_vector.h>
//#include <thrust/device_vector.h>
#include <thrust/generate.h>
#include <algorithm>
#include "utils.h"
#define MAX_THREADS_PER_BLOCK 512
// NOTE "scan" and "prefix-sum" (psum) are used interchangably in this context
__global__
void block_psum(const unsigned int * const g_in,
unsigned int * const g_out,
unsigned int * const g_sums,
const size_t n)
{
extern __shared__ unsigned int smem[];
const size_t bx = blockIdx.x * blockDim.x;
const size_t tx = threadIdx.x;
const size_t px = bx + tx;
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: ssh
[vbhamidipati1@cder01 ~]$ scp "/home/users/vbhamidipati1/HW3/psum.cu" vbhamidipati1@cder01:/home/users/vbhamidipati1/HW3
SSH: hpc_servers
Ln 9, Col 1 Spaces: 2 UTF-8 CRLF CUDA C++
Type here to search
2:06 AM
4/19/2020

(88).png


File Edit Selection View Go Run Terminal Help
psum.cu - vbhamidipati1 [SSH: hpc_servers] - Visual Studio Code
EXPLORER
OPEN EDITORS
stb_image.h HW3
psum.cu HW3
psum_hierarchical.cu HW3
nearest_neighbor_interpolation.c...
VBHAMIDIPATI1 [SSH: HPC_SERVERS]
HW3
colordefs.h
decimal.h
image.jpg
knn.h
nearest_neighbor_interpolation.cu
nearest_neighbor_interpolation.exe
output.png
psum_hierarchical.cu
psum_hierarchical.exe
psum.cu
psum.exe
sample.png
stb_image_write.h
stb_image.h
text.h
try.cu
try.exe
utils.h
wtime.h
ondemand / data / sys / dashboard / b...
.bash_history
.bash_logout
.bash_profile
OUTLINE
TIMELINE
stb_image.h
psum.cu
psum_hierarchical.cu
nearest_neighbor_interpolation.cu
HW3 > psum.cu
#include <thrust/host_vector.h>
//#include <thrust/device_vector.h>
#include <thrust/generate.h>
#include <algorithm>
#include "utils.h"
#define MAX_THREADS_PER_BLOCK 512
// NOTE "scan" and "prefix-sum" (psum) are used interchangably in this context
__global__
void block_psum(const unsigned int * const g_in,
unsigned int * const g_out,
unsigned int * const g_sums,
const size_t n)
{
extern __shared__ unsigned int smem[];
const size_t bx = blockIdx.x * blockDim.x;
const size_t tx = threadIdx.x;
const size_t px = bx + tx;
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: ssh
[vbhamidipati1@cder01 ~]$ nvcc ./HW3/psum.cu -o ./HW3/psum.exe
nvcc warning : The 'compute_20', 'sm_20', and 'sm_21' architectures are deprecated, and may be removed in a future release (Use -Wno-deprecated-gpu-targets to suppress warning).
[vbhamidipati1@cder01 ~]$
SSH: hpc_servers
Ln 9, Col 1 Spaces: 2 UTF-8 CRLF CUDA C++
Type here to search
2:06 AM
4/19/2020

(89).png



(90).png

Problem 2

(20 pts) Write a CUDA program to implement the single-pass scan version of the hierarchical prefix sum algorithm

Listing 4: Parallel Upsweep

```
__global__ void prefix_upsweep_kernel (int *b_d, int *a_d, int n, int depth, int *blocksum_device) {


    while (tid < n) {
        smem[threadIdx.x] = a_d[tid];       // each thread copy data to shared memory
        __syncthreads();                    // wait for all threads

        offset = 1;                         //1->2->4->8
        for (d = 1; d <= depth ; d++) {
            offset *= 2;
            if (threadIdx.x % offset == offset-1 ){
                smem[threadIdx.x]+= smem[threadIdx.x- offset/2];
                 __syncthreads();
            }

        } // end for loop
```

```
17
18          b_d[tid] = smem[threadIdx.x];        // *write the result to array↩
                b_d[tid] location
19
20          if (threadIdx.x == blockDim.x -1){
21               blocksum_device[blockIdx.x] = smem[threadIdx.x];
22          }
23
24          __syncthreads();                    // wait for all threads to ↩
                write results
25
26          tid += blockDim.x * gridDim.x;
27
28      } // end while (tid < n)
29  } // end kernel function
```

Following Listing 4...

Name: prefix_upsweep_kernel

Input: int *b_d, int *a_d, int n, int depth, int *blocksum_device

Output: int *blocksum_device

Operation: Performs the upsweep sum on each block. b_d is updated but not copied to CPU yet.

Note: Size of blocksum_device is same as number of blocks

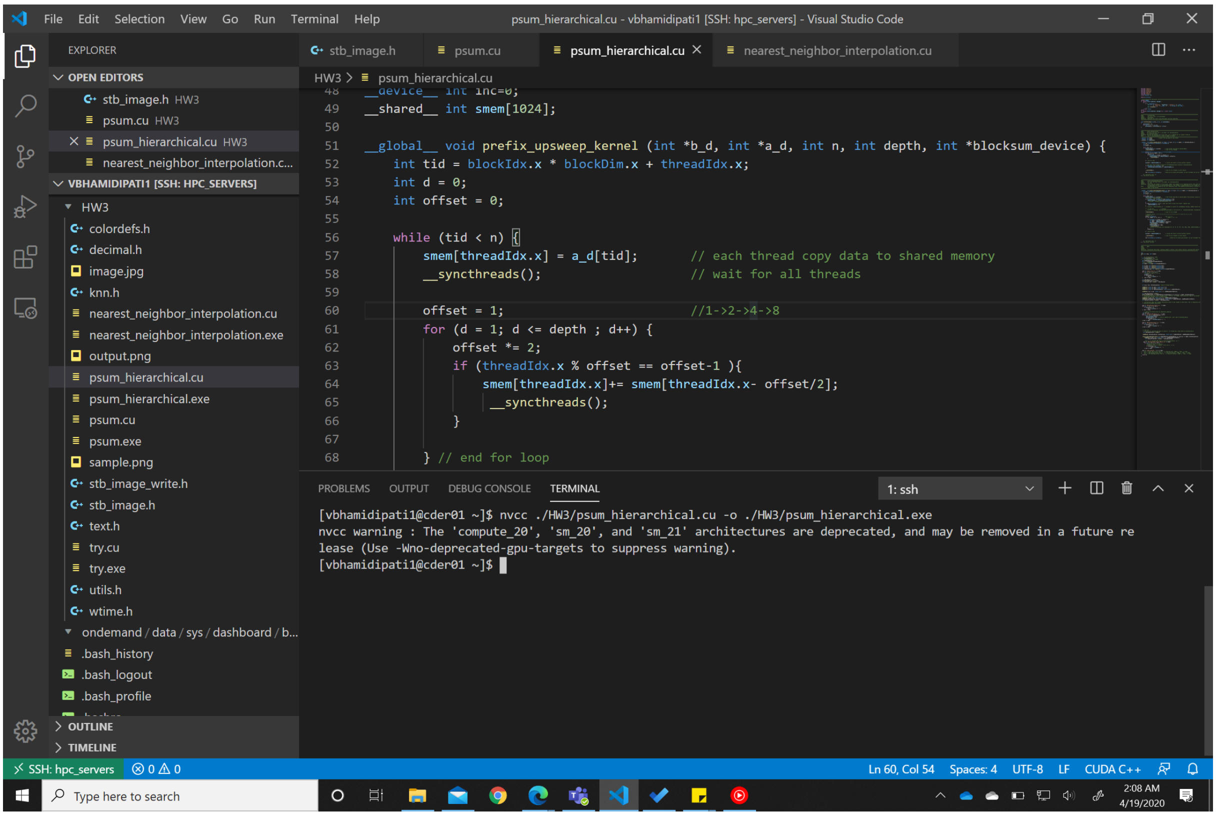
Listing 5: Parallel downsweep

```
1
2   __global__ void prefix_downsweepsweep_kernel (int *b_d, int *a_d, int n, ↩
        int depth, int *blocksum_device) {
3
4       while (tid < n) {
5           smem[threadIdx.x] = b_d[tid];       // each thread copy data to ↩
                shared memory from previous results b_d
6            //b_d[tid] = smem[threadIdx.x];
7           if (threadIdx.x ==  blockDim.x -1 && blockIdx.x != 0){
8               smem[threadIdx.x] = blocksum_device[blockIdx.x-1];
9                //b_d[tid] = blocksum_device[blockIdx.x-1];
10          }
11          if (tid == blockDim.x - 1 ){ // clear last entry in only first ↩
                block - special case
12               smem[threadIdx.x] = 0;
13          }
14
15          __syncthreads();                    // wait for all threads
16
17          offset = blockDim.x;                 //8 -> 4 -> 2
18          for (d = depth; d > 0 ; d--) {         // depth 3 -> 2->1
```

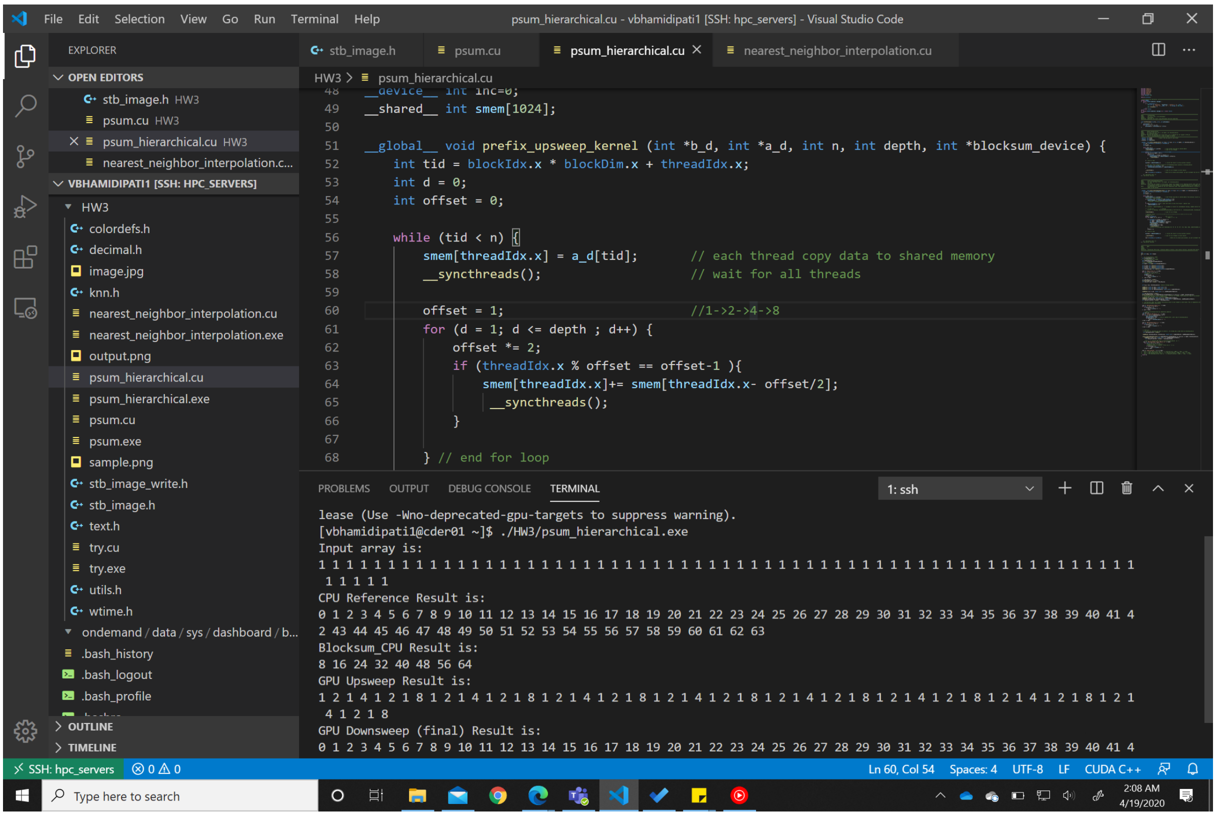
```
19
20              if (threadIdx.x % offset == offset-1 ){
21                  int tmp3 =  smem[threadIdx.x];
22                  int tmp1 =  smem[threadIdx.x- offset/2];
23                  smem[threadIdx.x- offset/2] = tmp3;
24                  __syncthreads();
25                  smem[threadIdx.x]+= tmp1;
26                   __syncthreads();
27              }
28              offset /= 2;
29
30          } // end for loop
31
32          b_d[tid] = smem[threadIdx.x];        // *write the result to array↩
                b_d[tid] location
33
34          __syncthreads();                    // wait for all threads to ↩
               write results
35
36          tid += blockDim.x * gridDim.x;
37
38      } // end while (tid < n)
39  } // end kernel function
```

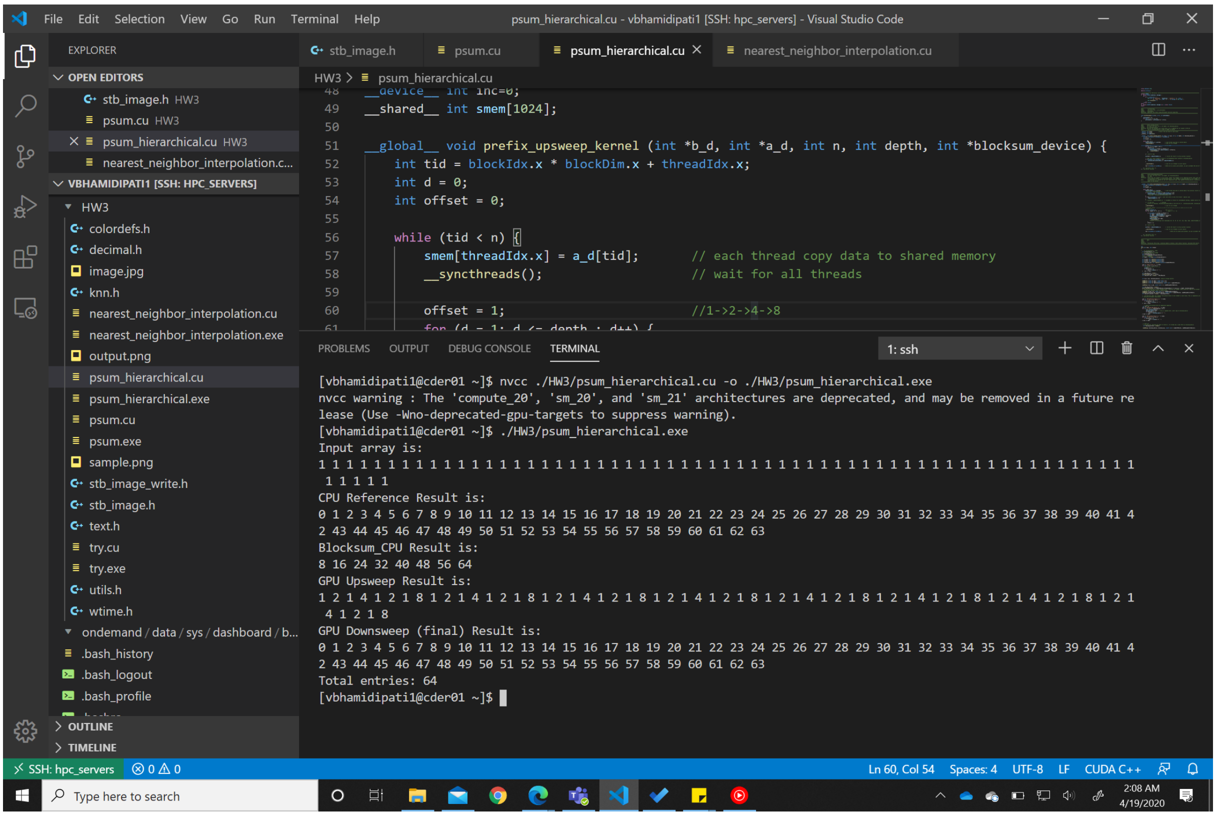
Following Listing 5...
Name: prefix_downsweepsweep_kernel
Input: int *b_d, int *a_d, int n, int depth, int *blocksum_device
Output: int *b_d
Operation: Clears the last element in first block. Copies last element in all remaining blocks (nth) with content from corresponding location in blocksum_device[n-1]. Last element in blocksum_device is not needed in exclusive scan.
Note: blocksum_device is updated by cpu with cummulative sums, then updated blocksum_device is given as input to this kernel

(91).png


psum_hierarchical.cu - vbhamidipati1 [SSH: hpc_servers] - Visual Studio Code
__shared__ int smem[1024];
__global__ void prefix_upsweep_kernel (int *b_d, int *a_d, int n, int depth, int *blocksum_device) {
int tid = blockIdx.x * blockDim.x + threadIdx.x;
int d = 0;
int offset = 0;
while (tid < n) {
smem[threadIdx.x] = a_d[tid]; // each thread copy data to shared memory
__syncthreads(); // wait for all threads
offset = 1; //1->2->4->8
for (d = 1; d <= depth ; d++) {
offset *= 2;
if (threadIdx.x % offset == offset-1){
smem[threadIdx.x]+= smem[threadIdx.x- offset/2];
__syncthreads();
}
} // end for loop
[vbhamidipati1@cder01 ~]$ nvcc ./HW3/psum_hierarchical.cu -o ./HW3/psum_hierarchical.exe
nvcc warning : The 'compute_20', 'sm_20', and 'sm_21' architectures are deprecated, and may be removed in a future re
lease (Use -Wno-deprecated-gpu-targets to suppress warning).
[vbhamidipati1@cder01 ~]$

(92).png


psum_hierarchical.cu - vbhamidipati1 [SSH: hpc_servers] - Visual Studio Code
lease (Use -Wno-deprecated-gpu-targets to suppress warning).
[vbhamidipati1@cder01 ~]$ ./HW3/psum_hierarchical.exe
Input array is:
CPU Reference Result is:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 4
2 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
Blocksum_CPU Result is:
8 16 24 32 40 48 56 64
GPU Upsweep Result is:
GPU Downsweep (final) Result is:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 4



(93).png

Problem 3

(30 pts)Write a CUDA program for doing nearest-neighbor interpolation. This pro-gram should receive as input an nxm image and the upsampling factork(an integer).Note that the resulting image will be kn x km and should look blocky.

Listing 6: Nearest Neighbor Resize

```
uint8_t *cudaNearestNeighborResize(uint8_t *origImageData,int origImgWidth,int origImgHeight,int newImgWidth,int newImgHeight,decimal_t resizeRatio)
{


    int blocksPerLine=ceil(floatDiv(newImgWidth,CUDA_THREADS_PER_BLOCK));
    int totalNumBlocks=blocksPerLine*newImgHeight; // Each block is responsible for a single line

    if(totalNumBlocks>65535)
        return 0;

    size_t origImageSize=origImgWidth*origImgHeight*sizeof(uint8_t);

    size_t newImageSize=newImgWidth*newImgHeight*sizeof(uint8_t);

```

```
15      cudaMalloc(&device_origImageData_in,origImageSize);
16      cudaMalloc(&device_newImageData_out,newImageSize);
17
18      cudaMemcpy(device_origImageData_in,origImageData,origImageSize,↩
            cudaMemcpyHostToDevice);
19
20      decimal_t origRatio=decimalDiv(1.0,decimal_t(resizeRatio));
21
22
23      device_nearestNeighborResize<<<totalNumBlocks,CUDA_THREADS_PER_BLOCK↩
            >>>(origImgWidth,newImgWidth,origRatio,blocksPerLine,↩
            device_origImageData_in,device_newImageData_out);
24
25      uint8_t *newImageData=(uint8_t*)malloc(newImageSize);
26      cudaMemcpy(newImageData,device_newImageData_out,newImageSize,↩
            cudaMemcpyDeviceToHost);
27
28      return newImageData;
29  }
```

Following Listing 6...
Nearest neighbor resize function resizes the image based on the upscaling factor. Copies the size of old image to the memory and creates space for new image based on the image size in the GPU. devixe_nearestNeighborResize takes the memory values into the CUDA device and performs resize operation

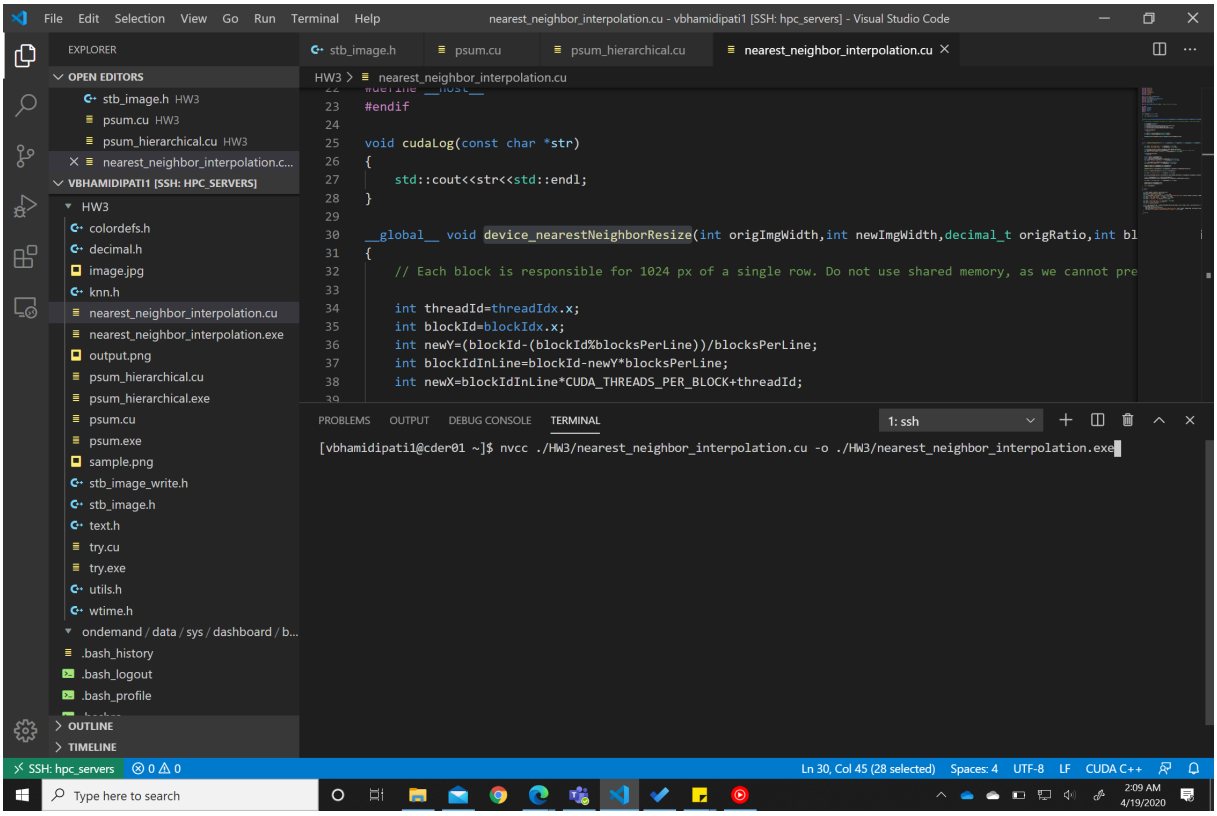
Listing 7: device nearest Neighbor Resize

```
1   __global__ void device_nearestNeighborResize(int origImgWidth,int ↩
        newImgWidth,decimal_t origRatio,int blocksPerLine,uint8_t *origImgData↩
        ,uint8_t *newImgData)
2   {
3       // Each block is responsible for 1024 px of a single row. Do not use ↩
            shared memory, as we cannot predict the distances between two ↩
            lines we need in the source image.
4
5       int newY=(blockId-(blockId%blocksPerLine))/blocksPerLine;
6       int blockIdInLine=blockId-newY*blocksPerLine;
7       int newX=blockIdInLine*CUDA_THREADS_PER_BLOCK+threadId;
8
9       if(newX>=newImgWidth)
10          return;
11
12      int oldX=(int)round(origRatio*((decimal_t)newX));
13      int oldY=(int)round(origRatio*((decimal_t)newY));
14
```

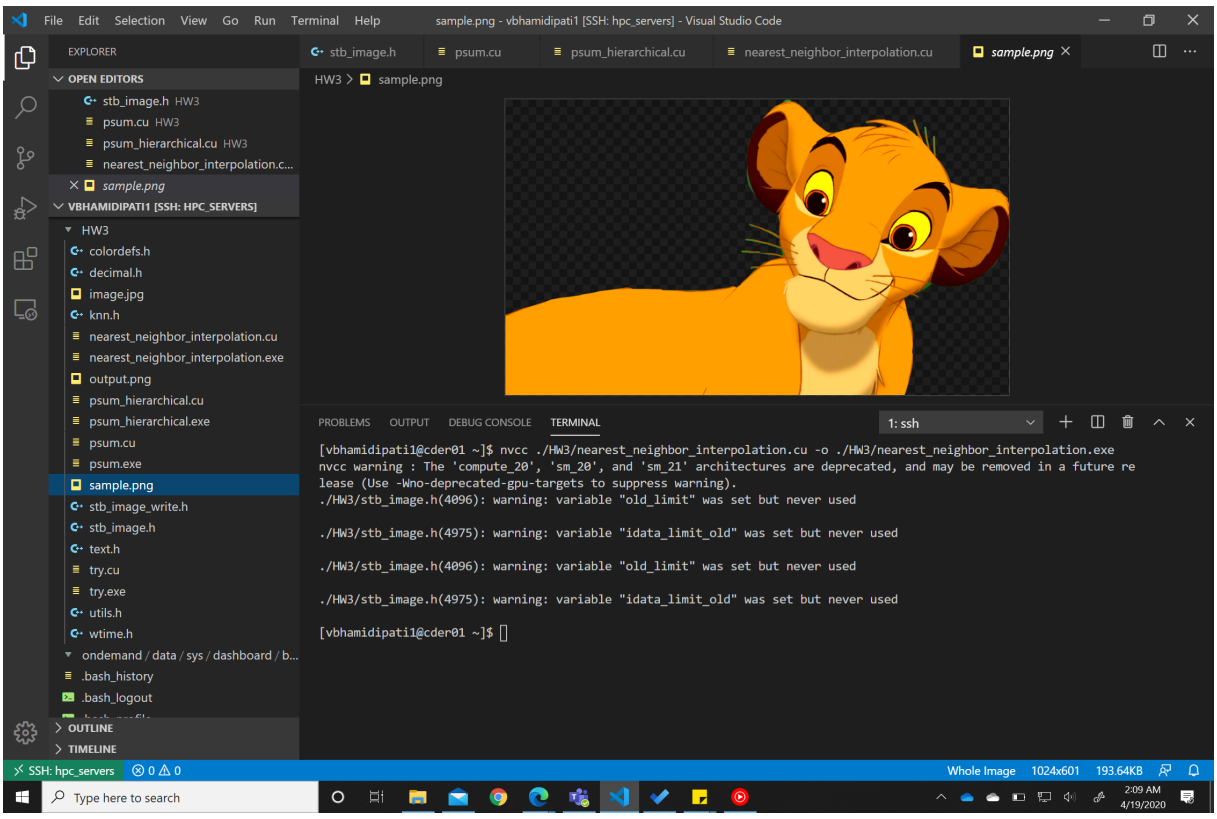
```
15      newImgData[newY*newImgWidth+newX]=origImgData[oldY*origImgWidth+oldX];
16
17  }
```

Following Listing 7...

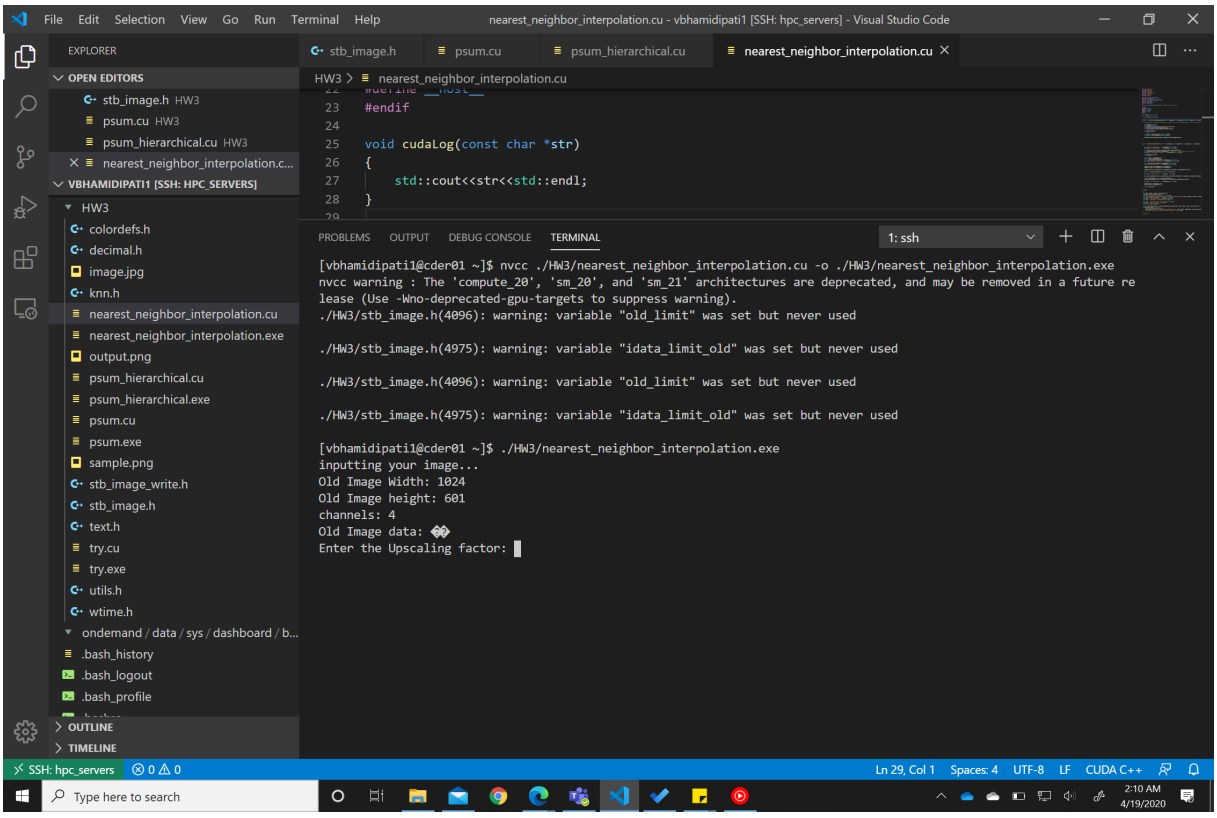
Based on the old image ratio the newer image is adjusted and new data containing the image data is retuned. Here, the image data is stored in the uint8_t which specifies that it is stored in unsigned int format of 8 bits.



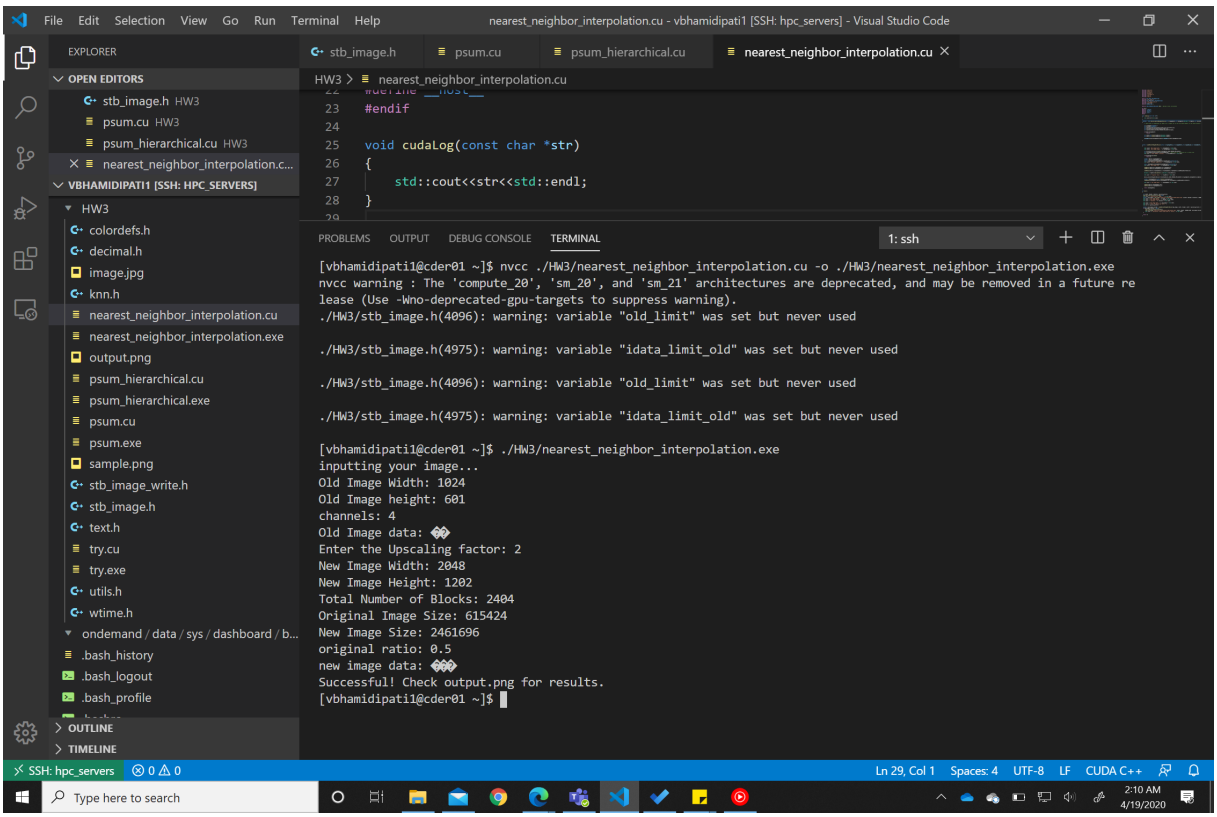
(94).png


File Edit Selection View Go Run Terminal Help
sample.png - vbhamidipati1 [SSH: hpc_servers] - Visual Studio Code
EXPLORER
OPEN EDITORS
stb_image.h HW3
psum.cu HW3
psum_hierarchical.cu HW3
nearest_neighbor_interpolation.c...
sample.png
VBHAMIDIPATI1 [SSH: HPC_SERVERS]
HW3
colordefs.h
decimal.h
image.jpg
knn.h
nearest_neighbor_interpolation.cu
nearest_neighbor_interpolation.exe
output.png
psum_hierarchical.cu
psum_hierarchical.exe
psum.cu
psum.exe
sample.png
stb_image_write.h
stb_image.h
text.h
try.cu
try.exe
utils.h
wtime.h
ondemand / data / sys / dashboard / b...
.bash_history
.bash_logout
OUTLINE
TIMELINE
stb_image.h
psum.cu
psum_hierarchical.cu
nearest_neighbor_interpolation.cu
sample.png
HW3 > sample.png
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: ssh
[vbhamidipati1@cder01 ~]$ nvcc ./HW3/nearest_neighbor_interpolation.cu -o ./HW3/nearest_neighbor_interpolation.exe
nvcc warning : The 'compute_20', 'sm_20', and 'sm_21' architectures are deprecated, and may be removed in a future release (Use -Wno-deprecated-gpu-targets to suppress warning).
./HW3/stb_image.h(4096): warning: variable "old_limit" was set but never used
./HW3/stb_image.h(4975): warning: variable "idata_limit_old" was set but never used
./HW3/stb_image.h(4096): warning: variable "old_limit" was set but never used
./HW3/stb_image.h(4975): warning: variable "idata_limit_old" was set but never used
[vbhamidipati1@cder01 ~]$
SSH: hpc_servers
0 0
Whole Image 1024x601 193.64KB
Type here to search
2:09 AM
4/19/2020

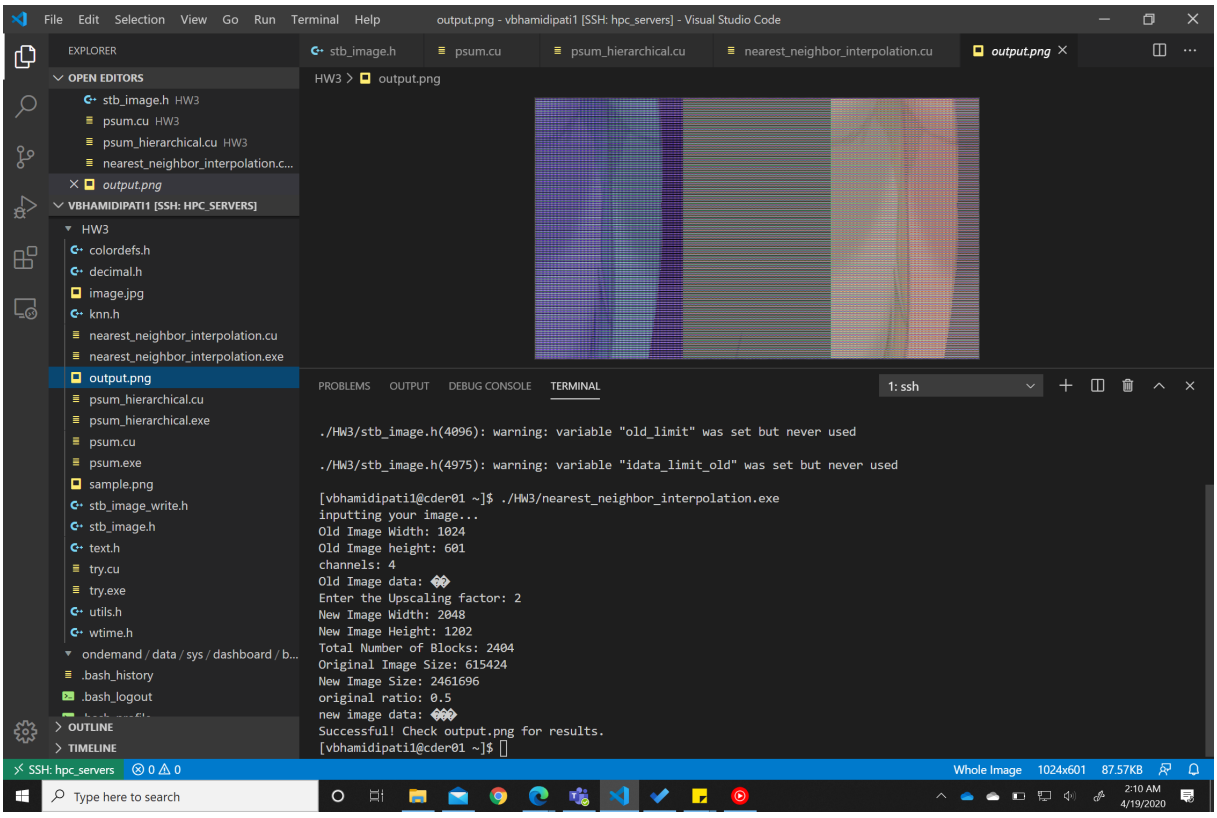
(95).png


File Edit Selection View Go Run Terminal Help
nearest_neighbor_interpolation.cu - vbhamidipati1 [SSH: hpc_servers] - Visual Studio Code
EXPLORER
OPEN EDITORS
stb_image.h HW3
psum.cu HW3
psum_hierarchical.cu HW3
nearest_neighbor_interpolation.c...
VBHAMIDIPATI1 [SSH: HPC_SERVERS]
HW3
colordefs.h
decimal.h
image.jpg
knn.h
nearest_neighbor_interpolation.cu
nearest_neighbor_interpolation.exe
output.png
psum_hierarchical.cu
psum_hierarchical.exe
psum.cu
psum.exe
sample.png
stb_image_write.h
stb_image.h
text.h
try.cu
try.exe
utils.h
wtime.h
ondemand / data / sys / dashboard / b...
.bash_history
.bash_logout
.bash_profile
OUTLINE
TIMELINE
stb_image.h
psum.cu
psum_hierarchical.cu
nearest_neighbor_interpolation.cu
HW3 > nearest_neighbor_interpolation.cu
#endif
void cudaLog(const char *str)
{
std::cout<<str<<std::endl;
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: ssh
[vbhamidipati1@cder01 ~]$ nvcc ./HW3/nearest_neighbor_interpolation.cu -o ./HW3/nearest_neighbor_interpolation.exe
nvcc warning : The 'compute_20', 'sm_20', and 'sm_21' architectures are deprecated, and may be removed in a future release (Use -Wno-deprecated-gpu-targets to suppress warning).
./HW3/stb_image.h(4096): warning: variable "old_limit" was set but never used
./HW3/stb_image.h(4975): warning: variable "idata_limit_old" was set but never used
./HW3/stb_image.h(4096): warning: variable "old_limit" was set but never used
./HW3/stb_image.h(4975): warning: variable "idata_limit_old" was set but never used
[vbhamidipati1@cder01 ~]$ ./HW3/nearest_neighbor_interpolation.exe
inputting your image...
Old Image Width: 1024
Old Image height: 601
channels: 4
Old Image data: ??
Enter the Upscaling factor:
SSH: hpc_servers
0 0
Ln 29, Col 1 Spaces: 4 UTF-8 LF CUDA C++
Type here to search
2:10 AM
4/19/2020

(96).png



(97).png



(98).png

Program 1 -> psum.cu

Program 2 -> psum_hierarchical.cu

Program 3 -> nearest_neighbor_interpolation.cu