Лекция 13

- **Вычислительные шейдеры (продолжение).**

util_template.cpp

```
GLuint bufferID[2];
void csMove(GLuint* , int );
 int initBuffer(){
   glGenBuffers( 2,bufferID);
   glBindBuffer( GL_ARRAY_BUFFER, bufferID[0]);
   glBufferData( GL_ARRAY_BUFFER, 6*num_of_verticies*sizeof(float),
                                                    0, GL_DYNAMIC_DRAW );
   glBindBuffer( GL_ARRAY_BUFFER, bufferID[1]);
   glBufferData( GL_ARRAY_BUFFER, 3*num_of_verticies*sizeof(float),
                                                     0, GL_DYNAMIC_DRAW );
   csDataInit(bufferID, num_of_verticies);
   return 0;
}
void display(){
……
   csMove(bufferID, num_of_verticies);
}
```

```
void csDataInit(GLuint* inBuf,int N){
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 0, inBuf[0]);
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 1, inBuf[1]);

   GLuint computeShaderID=genComputeProg();
    glUseProgram(computeShaderID);
    glDispatchCompute(N/128, 1, 1);

    glMemoryBarrier(GL_SHADER_STORAGE_BARRIER_BIT);
}
```

csh_template.cpp

```
GLuint genComputeProg(){
   GLuint progHandle = glCreateProgram();
   GLuint cs = glCreateShader(GL_COMPUTE_SHADER);

   const char *cpSrc[] = {
      "#version 430\n",
      "layout (local_size_x = 128, local_size_y = 1, local_size_z = 1) in; \
       layout(std430, binding = 0) buffer PositionBuffer{float Pos[];};\
       layout(std430, binding = 1) buffer VelocityBuffer{float Vel[];};\
       float lmap(in uint i){\
          uint count;\
          float x=0.78;\
          for(count=0;count<i;count++)\
          x=3.99*x*(1-x);\
          return x;\
       }\
```

```
void main() {\
   uint index = gl_GlobalInvocationID.x;\
   Pos[index*6]=-0.5+1.0*lmap(index);\
   Pos[index*6+1]=-0.5+1.0*lmap(index*10);\
   Pos[index*6+2]=0.0;\
   Pos[index*6+3]=1.0;\
   Pos[index*6+4]=1.0;\
   Pos[index*6+5]=0.0;\
   Vel[3*index]=-0.5+1.0*lmap(index);\
   Vel[3*index+1]=-0.5+1.0*lmap(index*10);\
   Vel[3*index+2]=0.0;\
}"
};
```

……

csh_move.cpp

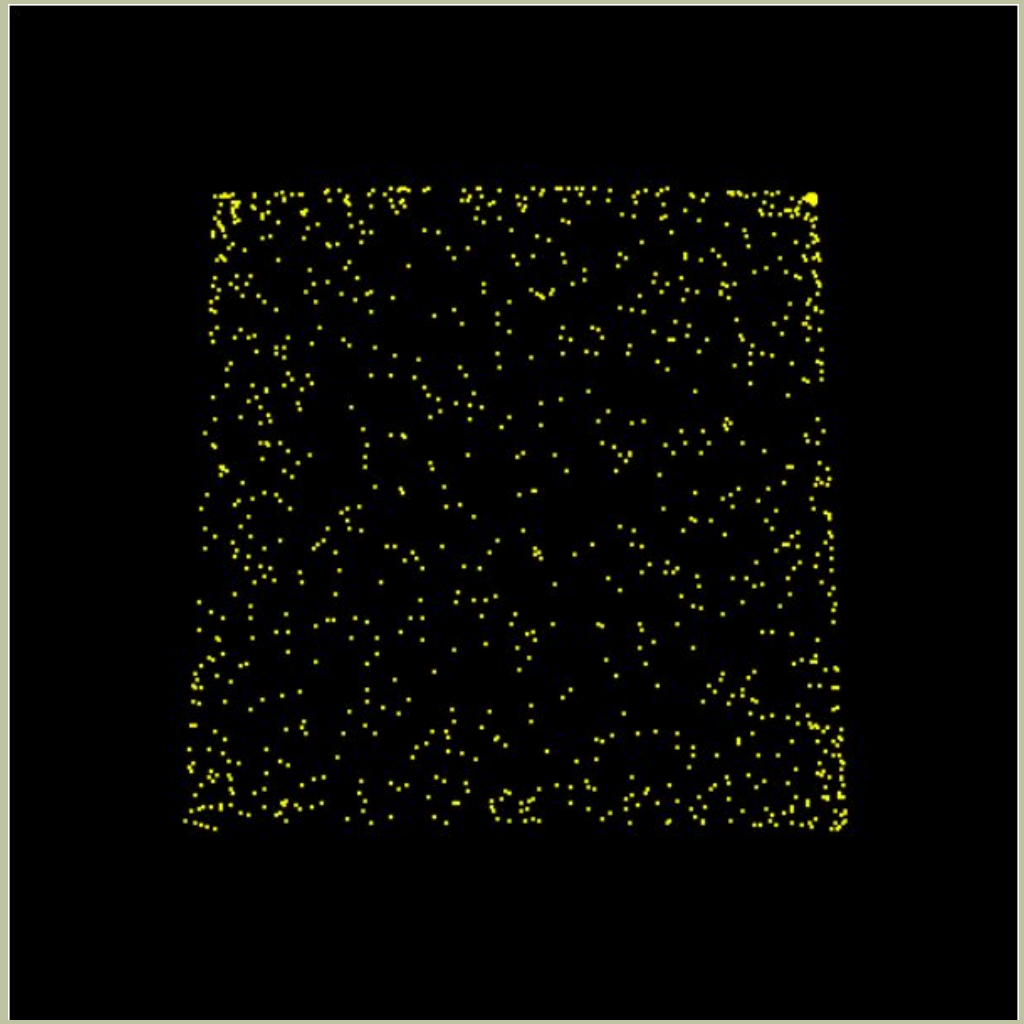
```
GLuint genMoveProg();
void csMove(GLuint* inBuf,int N){
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 0, inBuf[0]);
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 1, inBuf[1]);

   GLuint computeShaderID=genMoveProg();
   glUseProgram(computeShaderID);
   glDispatchCompute(N/128, 1, 1);
   glMemoryBarrier( GL_SHADER_STORAGE_BARRIER_BIT);
}
```

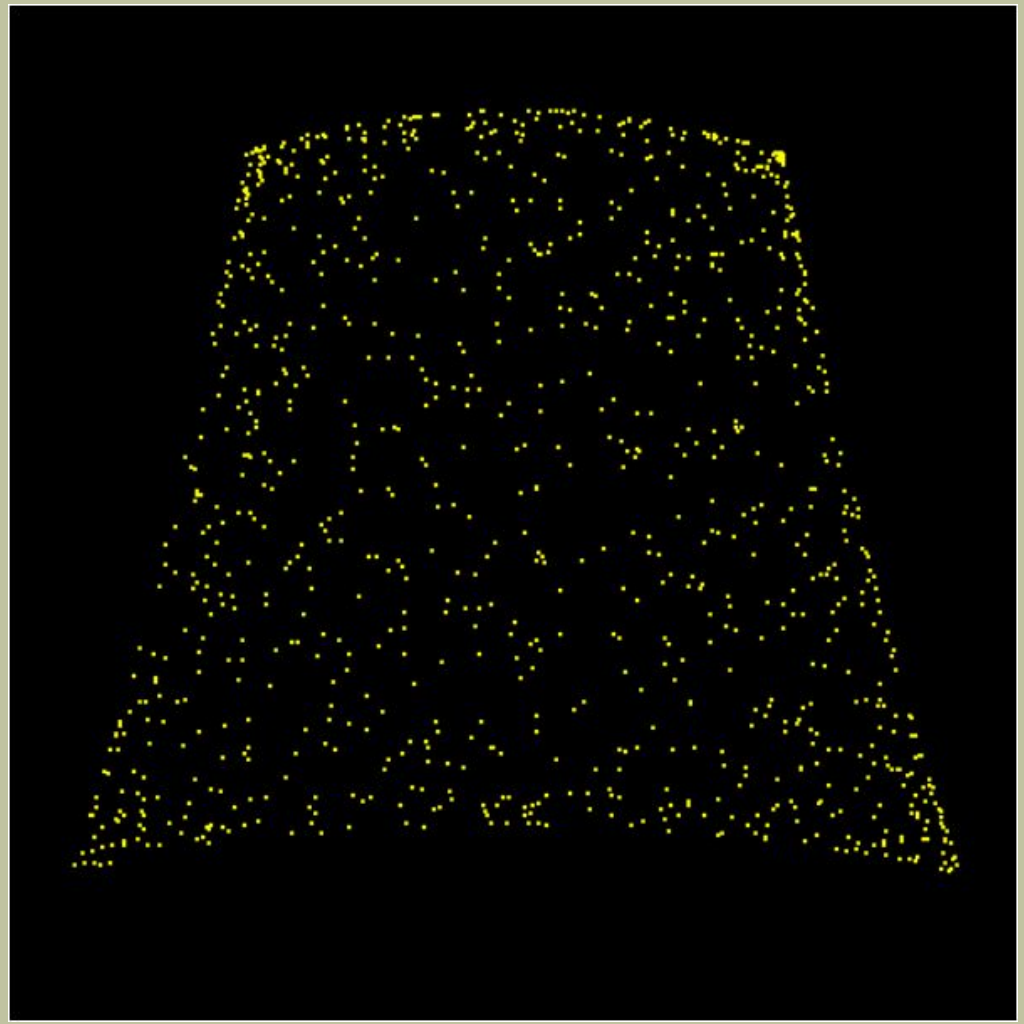
```
GLuint genMoveProg(){
   GLuint progHandle = glCreateProgram();
   GLuint cs = glCreateShader(GL_COMPUTE_SHADER);

   const char *cpSrc[] = {
      "#version 430\n",
      "layout (local_size_x = 128, local_size_y = 1, local_size_z = 1) in; \
       layout(std430, binding = 0) buffer PositionBuffer{float Pos[];};\
       layout(std430, binding = 1) buffer VelocityBuffer{float Vel[];};\
```

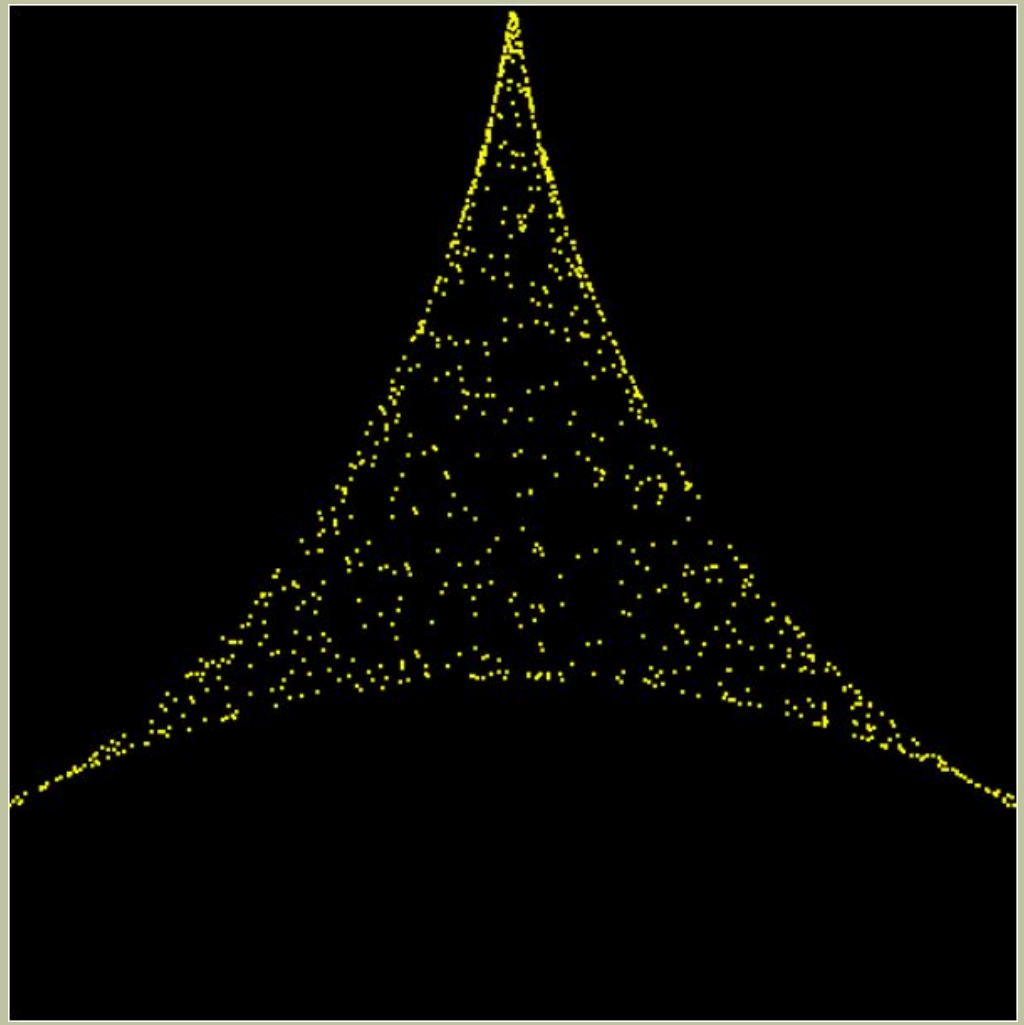
```
void main() {\
   float x,y,vx,vy;\
   float tau=0.01;\
   float c=2.0;\
   float eps=0.1;\
   uint index = gl_GlobalInvocationID.x;\
   x=Pos[index*6];\
   y=Pos[index*6+1];\
   vx=Vel[3*index];\
   vy=Vel[3*index+1];\
\
```

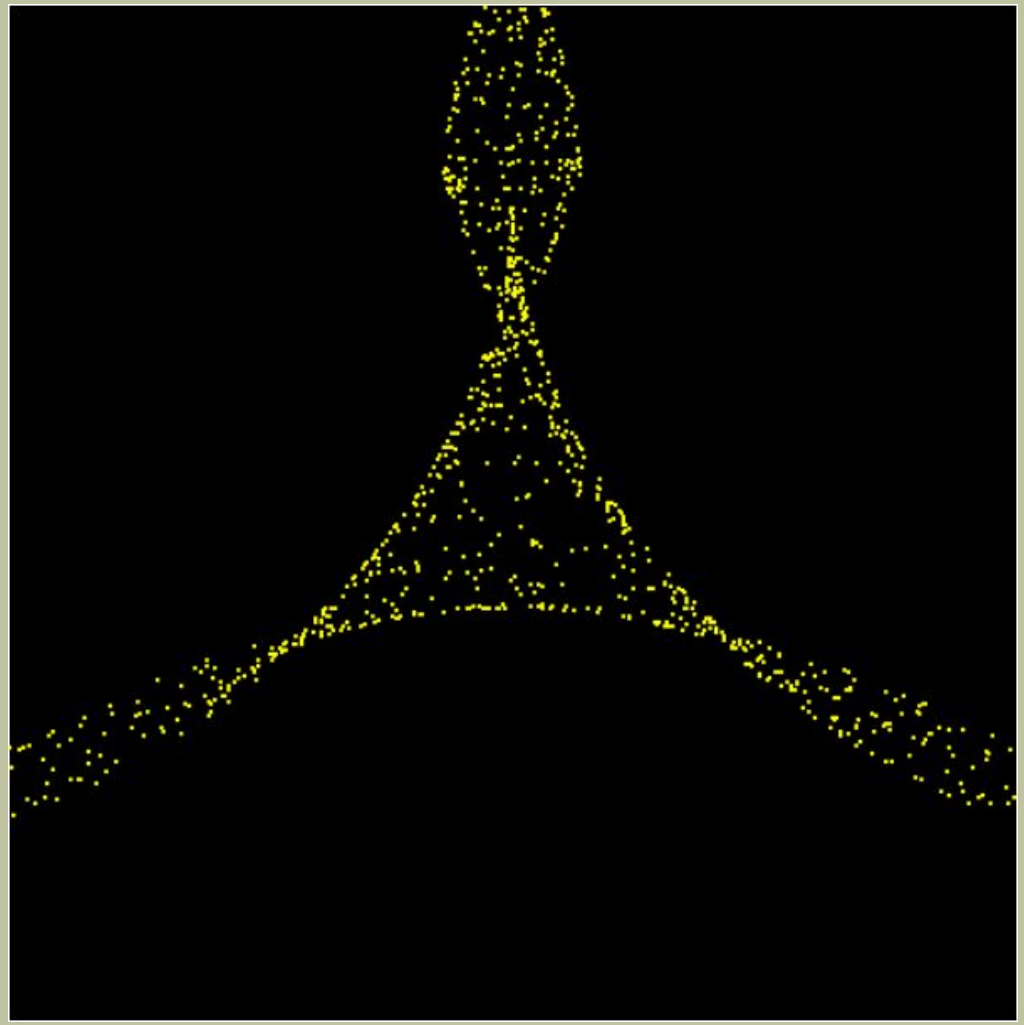
```
  vx=vx+tau*(-x-eps*(2*x*y));\
  vy=vy+tau*(-y-eps*(x*x-y*y));\
  x=x+tau*vx;\
  y=y+tau*vy;\
  \
  Pos[index*6]=x;\
  Pos[index*6+1]=y;\
  Vel[3*index]=vx;\
  Vel[3*index+1]=vy;\
 }"
};
```

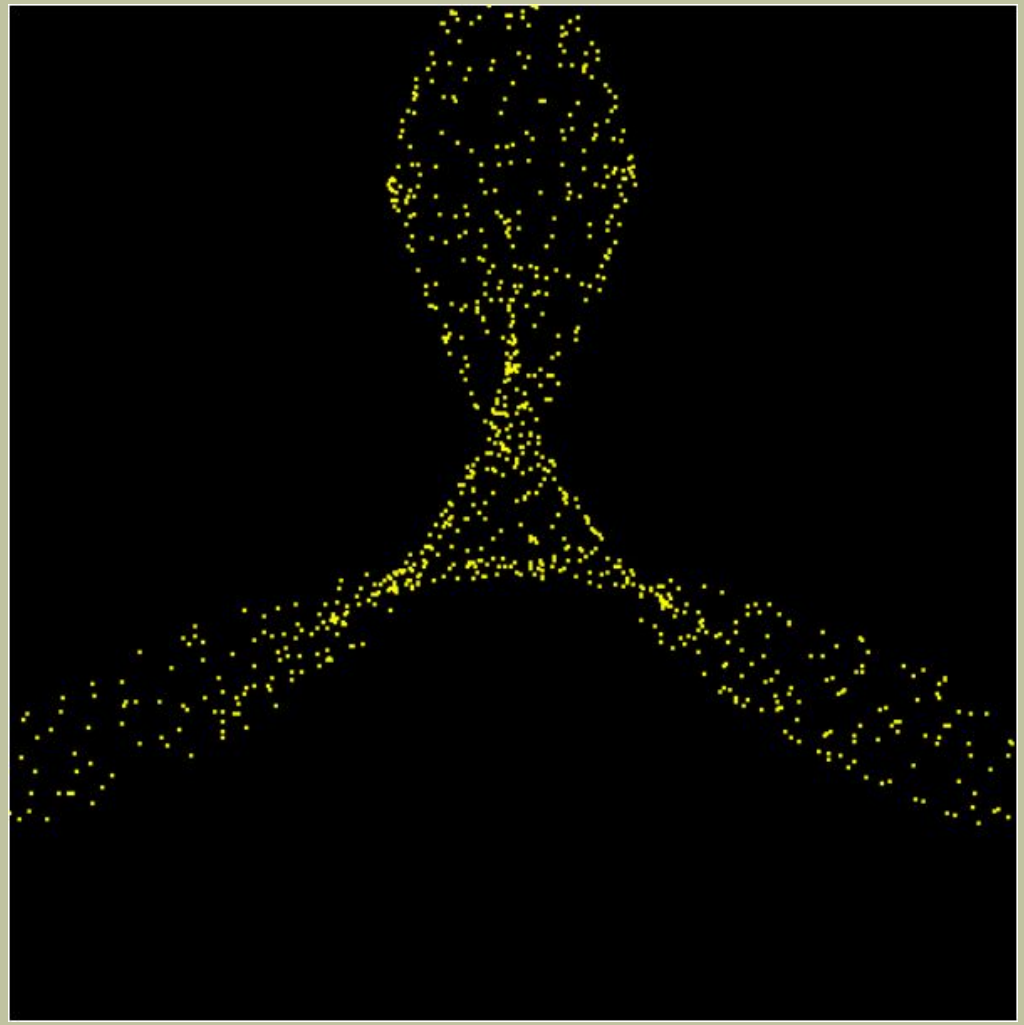
.....

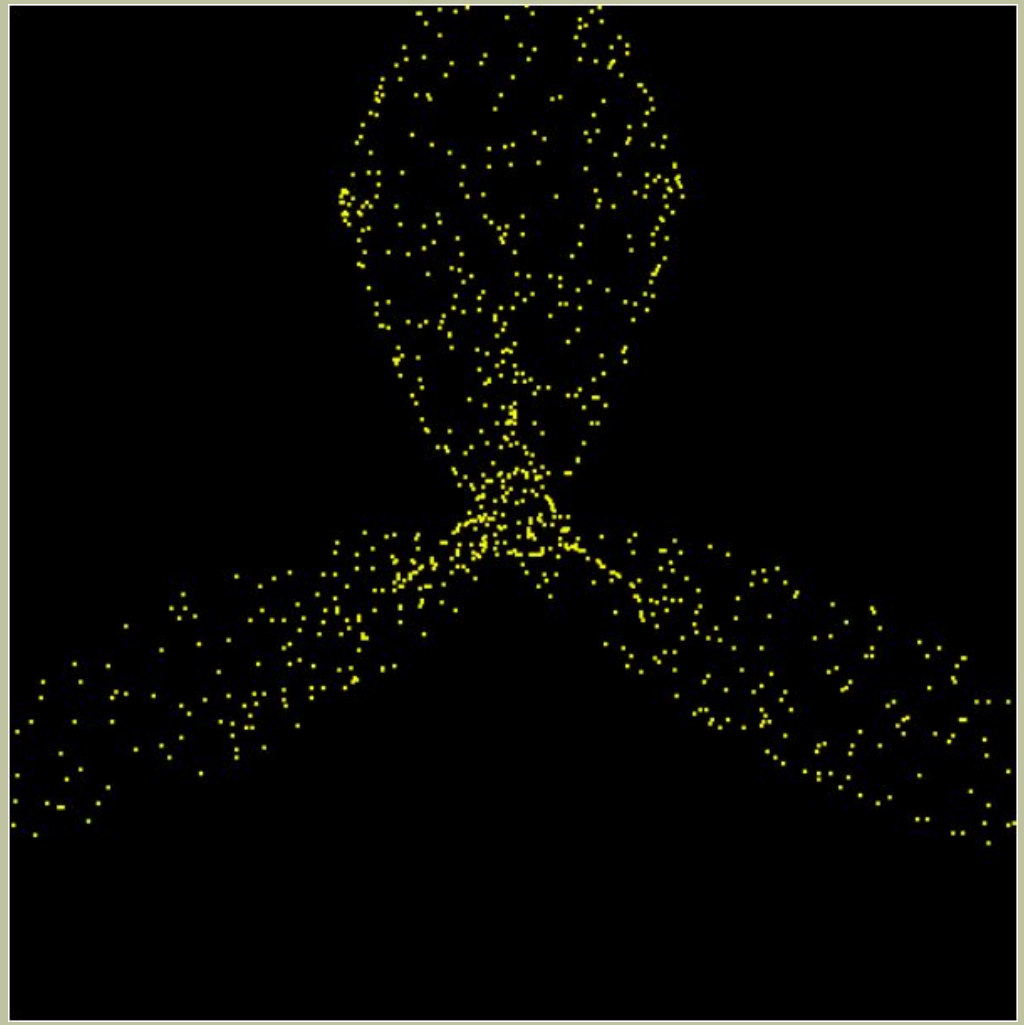


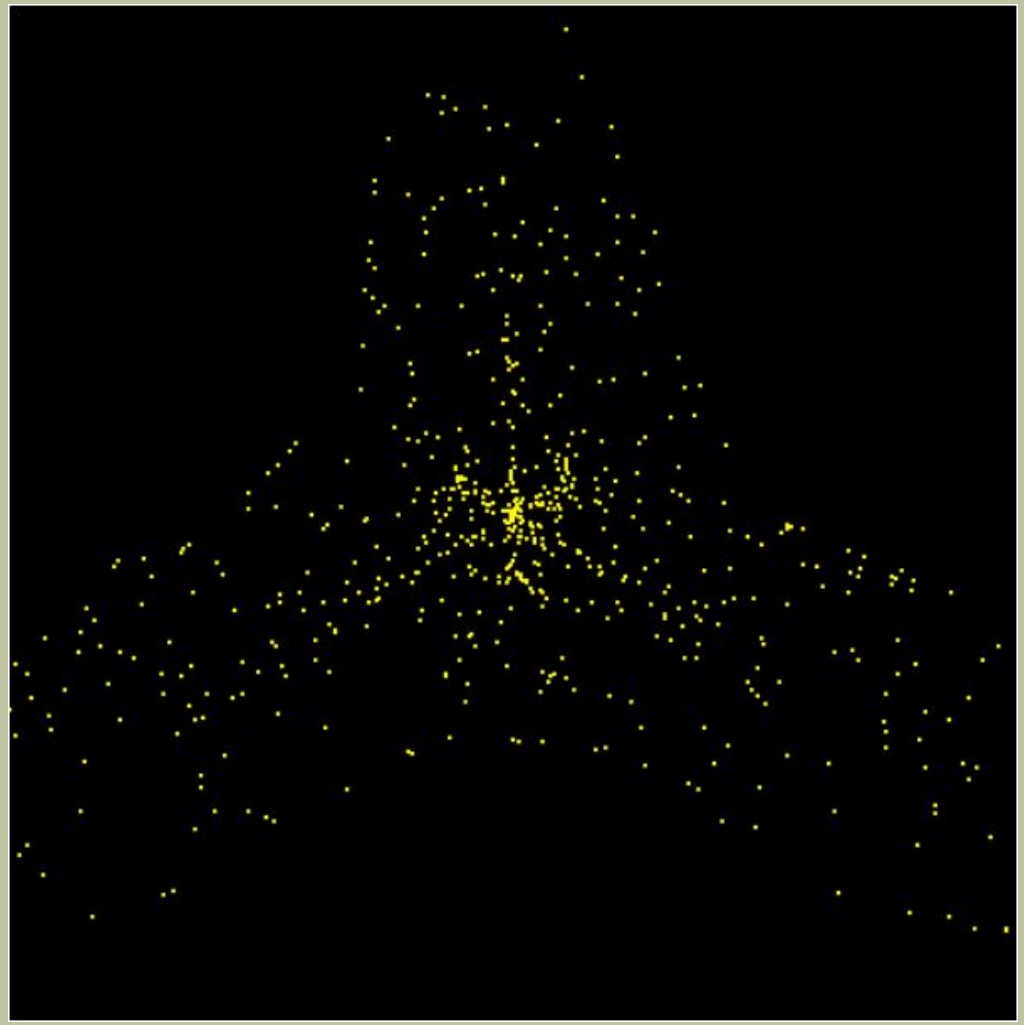


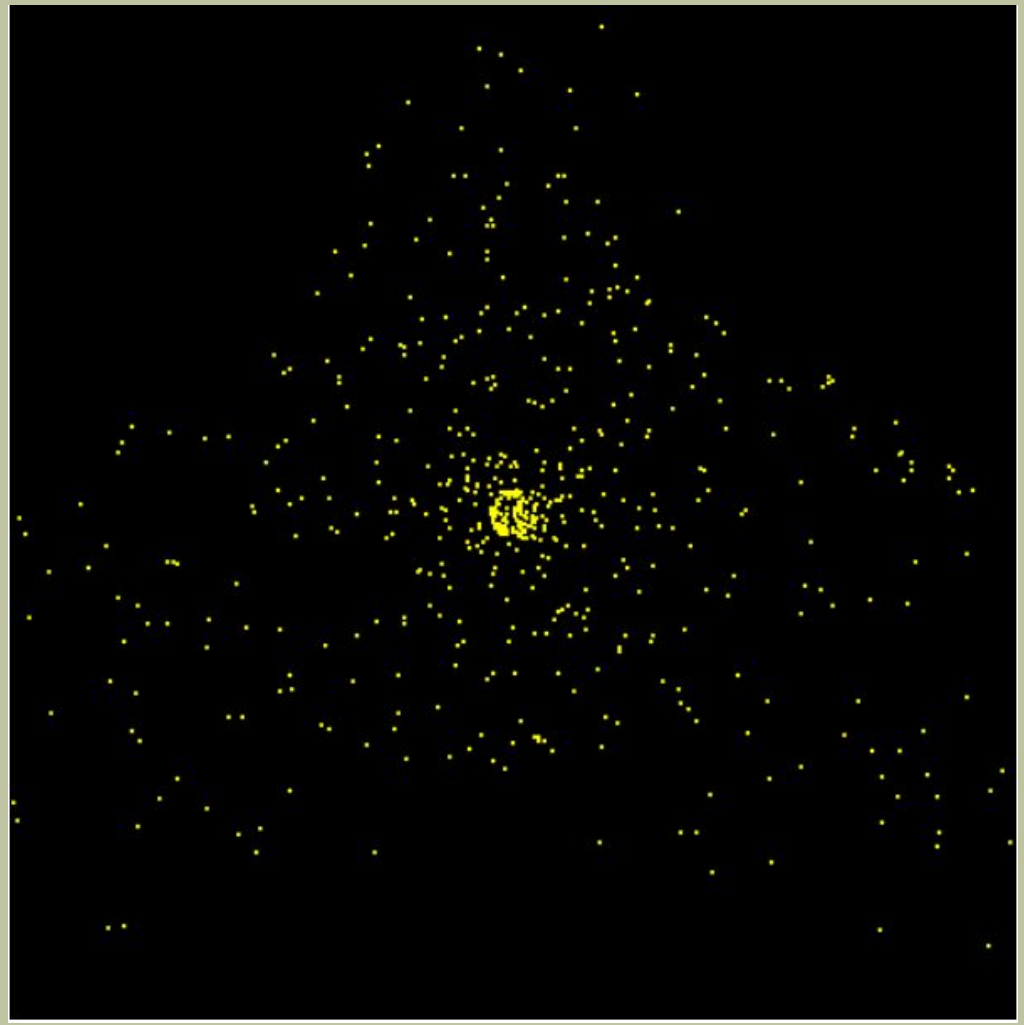


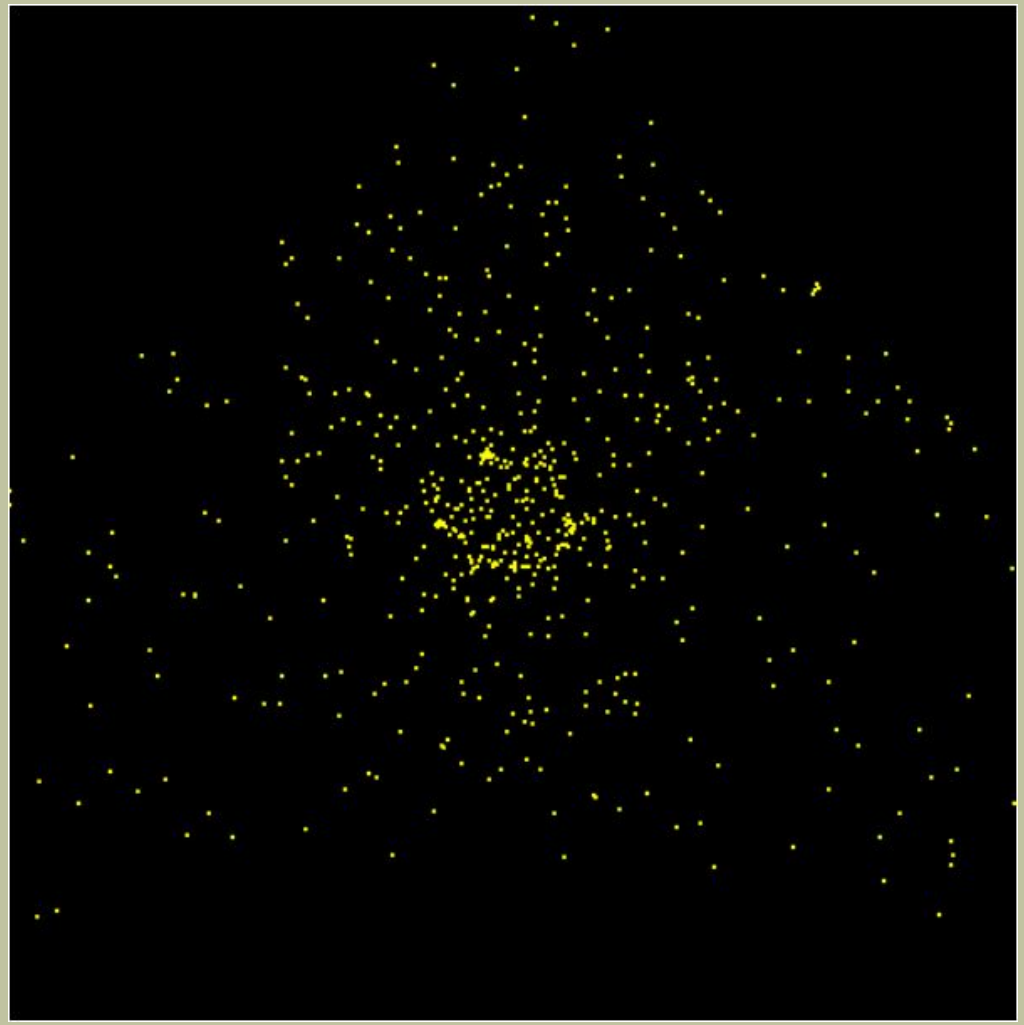


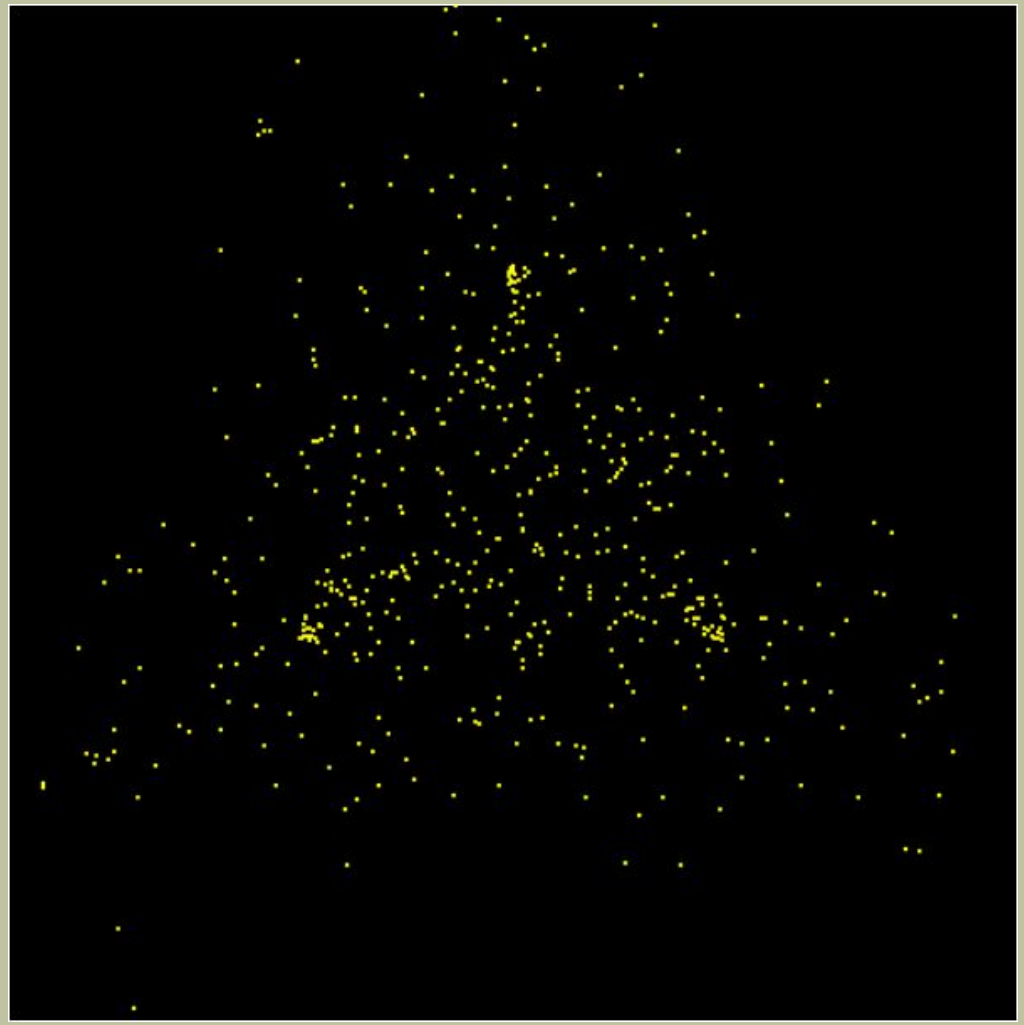












Вычислительные шейдеры — вычисления общего назначения

main.cpp

```
#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <stdio.h>
#include <malloc.h>

const unsigned int window_width = 512;
const unsigned int window_height = 512;
void initGL();

GLuint* bufferID;
void initBuffers(GLuint*&);
void transformBuffers(GLuint*);
void outputBuffers(GLuint*);
```

```
int main(){
   initGL();

   bufferID=(GLuint*)calloc(2, sizeof(GLuint));

   initBuffers(bufferID);
   transformBuffers(bufferID);
   outputBuffers(bufferID);

    glDeleteBuffers(2,bufferID);
    free(bufferID);
    glfwTerminate();

    return 0;
}
```

```
void initGL(){
  GLFWwindow*window;
……..
  glfwWindowHint(GLFW_VISIBLE, 0);
……
  window = glfwCreateWindow( window_width, window_height, "Template
                                                    window", NULL, NULL);

  glfwMakeContextCurrent(window);
……
  glewInit();
}
```

csh_common.cpp

```
const int N=256;
GLuint genInitProg();
int initBuffers(GLuint*& bufferID){
   glGenBuffers(2, bufferID);
   glBindBuffer(GL_SHADER_STORAGE_BUFFER, bufferID[0]);
   glBufferData(GL_SHADER_STORAGE_BUFFER, N * sizeof(float), 0,
                                                  GL_DYNAMIC_DRAW);
   glBindBuffer(GL_SHADER_STORAGE_BUFFER, bufferID[1]);
   glBufferData(GL_SHADER_STORAGE_BUFFER, N * sizeof(float), 0,
                                                   GL_DYNAMIC_DRAW);
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 0, bufferID[0]);
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 1, bufferID[1]);
```

```
  GLuint csInitID=genInitProg();
  glUseProgram(csInitID);
  glDispatchCompute(N/128, 1, 1);
  glMemoryBarrier(GL_SHADER_STORAGE_BARRIER_BIT |
                                        GL_BUFFER_UPDATE_BARRIER_BIT);
  glDeleteProgram(csInitID);
}
```

```
GLuint genInitProg(){
   GLuint progHandle = glCreateProgram();
   GLuint cs = glCreateShader(GL_COMPUTE_SHADER);
   const char *cpSrc[] = {
       "#version 430\n",
       "layout (local_size_x = 128, local_size_y = 1, local_size_z = 1) in; \
        layout(std430, binding = 0) buffer BufferA{float A[];};\
        layout(std430, binding = 1) buffer BufferB{float B[];};\
        void main() {\
           uint index = gl_GlobalInvocationID.x;\
           A[index]=0.1*float(index);\
           B[index]=0.2*float(index);\
   }"
};
```

.....

```
GLuint genTransformProg();
int transformBuffers(GLuint* bufferID){
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 0, bufferID[0]);
   glBindBufferBase(GL_SHADER_STORAGE_BUFFER, 1, bufferID[1]);

   GLuint csTransformID=genTransformProg();
   glUseProgram(csTransformID);
   glDispatchCompute(N/128, 1, 1);
   glMemoryBarrier(GL_SHADER_STORAGE_BARRIER_BIT |
                                          GL_BUFFER_UPDATE_BARRIER_BIT);

   glDeleteProgram(csTransformID);
}
```

```
GLuint genTransformProg(){
   GLuint progHandle = glCreateProgram();
   GLuint cs = glCreateShader(GL_COMPUTE_SHADER);

    const char *cpSrc[] = {
        "#version 430\n",
        "layout (local_size_x = 128, local_size_y = 1, local_size_z = 1) in; \
         layout(std430, binding = 0) buffer BufferA{float A[];};\
         layout(std430, binding = 1) buffer BufferB{float B[];};\
         void main() {\
             uint index = gl_GlobalInvocationID.x;\
             A[index]=A[index]+B[index];\
         }"
    };
```

.....

Транспонирование матриц (наивный вариант)

```
"#version 430\n",
"layout (local_size_x = 16, local_size_y = 16, local_size_z = 1) in; \
 layout(std430, binding = 0) buffer BufferA{float A[];};\
 layout(std430, binding = 1) buffer BufferB{float B[];};\
 uniform float coeff;\
    void main() {\
       uint indexX = gl_GlobalInvocationID.x;\
       uint indexY = gl_GlobalInvocationID.y;\
       A[indexX+16*indexY]=float(indexX+16*indexY);\
       barrier();\
       B[indexX+16*indexY]=A[indexY+16*indexX] + coeff;\
    }"
```

```
int transformBuffers(GLuint* bufferID){
 ……
  glUseProgram(csTransformID);

  GLuint coeffID = glGetUniformLocation(csTransformID,"coeff");
  float c=2.5;
  glUniform1f(coeffID, c);

  glDispatchCompute(Nx/16, Ny/16, 1);
……
}
```

Экспорт данных из буфера на хост

```
void outputBuffers(GLuint* bufferID){
   glBindBuffer(GL_SHADER_STORAGE_BUFFER, bufferID[1]);
   float* data = (float*)glMapBuffer(GL_SHADER_STORAGE_BUFFER,
                                                    GL_READ_ONLY);
   float* hdata=(float*)calloc(N, sizeof(float));
   memcpy(&hdata[0], data, sizeof(float)*N);
   glUnmapBuffer(GL_SHADER_STORAGE_BUFFER);

   for(int i = 0; i < Nx; i++){
      for(int j = 0; j < Ny; j++)
         fprintf(stdout,"%g\t",hdata[j+i*Ny]);
      printf("\n");
   }
}
```

```
> ./lab13b
2.5    18.5    34.5……    178.5   194.5   210.5   226.5   242.5
3.5    19.5    35.5……    179.5   195.5   211.5   227.5   243.5
4.5    20.5    36.5……    180.5   196.5   212.5   228.5   244.5
……..
15.5   31.5    47.5……    191.5   207.5   223.5   239.5   255.5
16.5   32.5    48.5……    192.5   208.5   224.5   240.5   256.5
17.5   33.5    49.5……    193.5   209.5   225.5   241.5   257.5
```