Introducing Dynamic Walls into Integer Lattice Gas Simulations

David Jedynak

NDSU

May 10th, 2018

Introduction

- Explore interactions between gas and rigid shapes
- Build off of existing Lattice Gas Simulation Code

initial goals

- Non leaking dynamic walls
- Make complex shapes out of these dynamic walls
- reproduce the Feynman tube experiment.

Method 1

Expected value of flow

$$< flow >= \text{particle density} * \text{wall velocity}$$

$$0 < flow < \text{min particle density}$$

```
    int tmp_0 = n[vx+x][y+vy][8-v];
int tmp = n[vx+x][y+vy][v];
    int max_random = 1;

    if(tmp > tmp_0){ max_random = tmp_0;}

    else{ max_random = tmp;}

    if(max_random > 0){ flow = (rand()%max_rand

    else{ flow = 0;}

n[x+vx][((y+vy))][v] = n[x][y][8-v] - flow;
n[x][y][8-v] = tmp + flow;
```
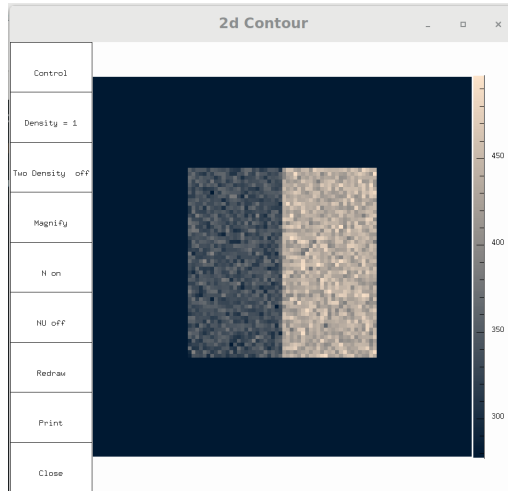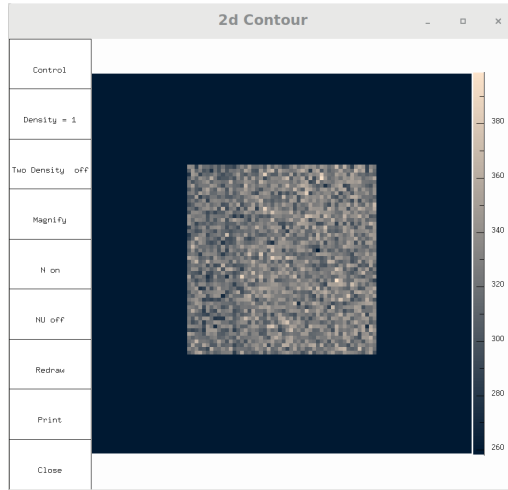
Results
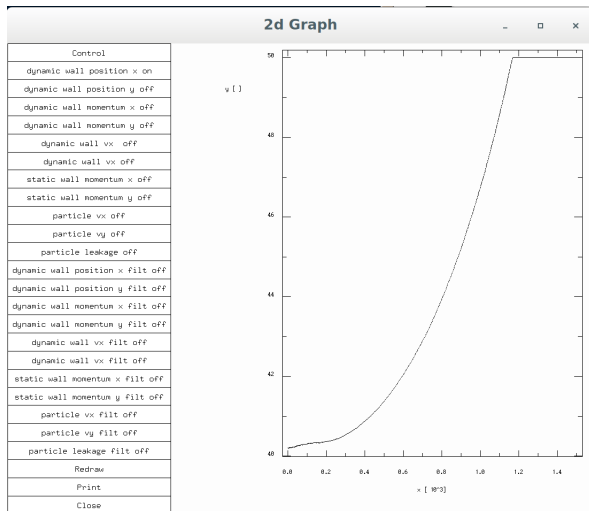

Figure 1:

¡text¿

Results

Figure 2:

¡text¿

Results

Figure 3:
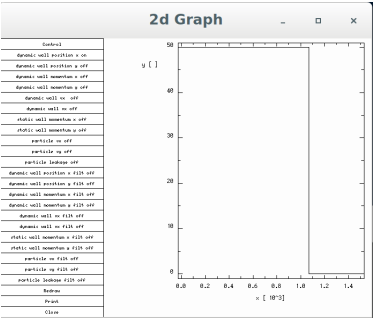
¡text¿

Results

Figure 4:

¡text¿

Results

Figure 5:

¡text¿
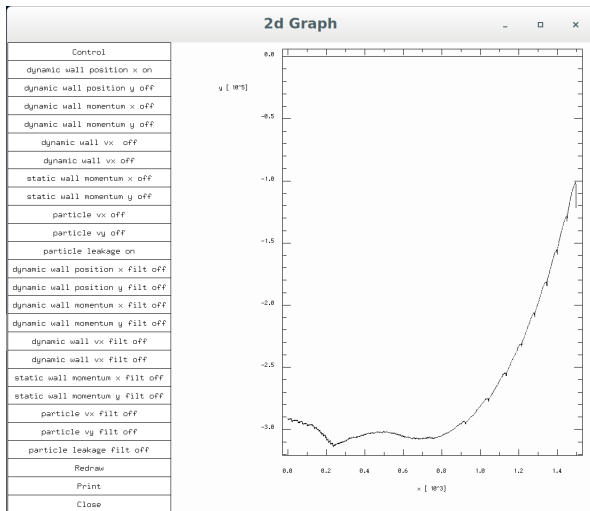
Method 2

In more detail, the probability that

$$pr * \text{particle density}$$

number of particles will be moved is

$$pr = \frac{\text{Wall Vx}}{1 - (\text{real(Wall x) - int(Wall x)})}$$

.

```
int tmp = n[x_v_b][y_v_b][v];
int iwp = dynamic_wall_position_x;// integer wall po
int flow = 0;// particles to be moved
double pr = dynamic_wall_vx/(1-(dynamic_wall_positio
if (rand()%1000 <= 1000*pr){
        flow = pr*n[x_b][y_b][v];
        n[x_b+1][y_b][v] +=flow;
        n[x_b][y_b][v] -=flow;}

n[x_v_b][y_v_b][v] = n[x_b][y_b][8-v];
n[x_b][y_b][8-v] = tmp;// + flow;
```
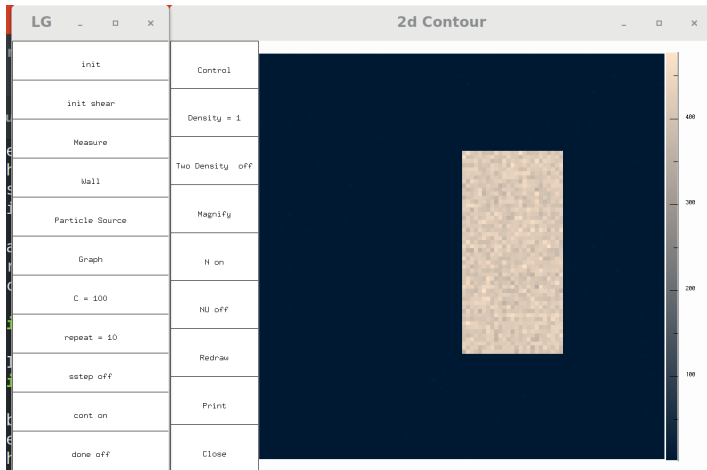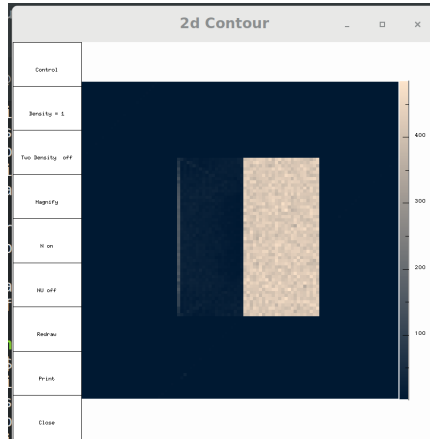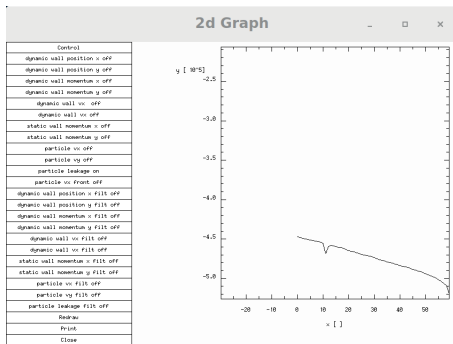
Results

Figure 6:

¡text¿

Results

Figure 7:

¡text¿

Results
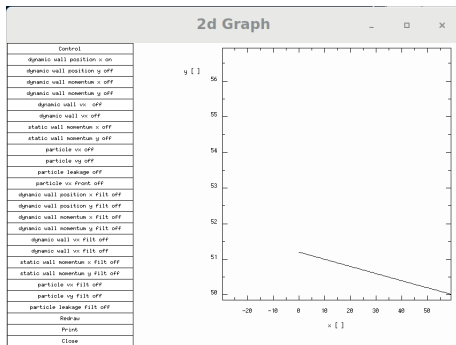

Figure 8:

¡text¿

Results

Figure 9:

¡text¿

Method 3

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_i)}{\partial x_i} = \nabla(\rho) + \upsilon * \nabla(\nabla(U) + (\nabla(U)T)) \qquad (1)$$

The partial for ρ and ρu_i can be set to zero. This gives us:

$$0 = \nabla(\rho) + \upsilon * \nabla(\nabla(U) + (\nabla(U)T)) \qquad (2)$$

$$\nabla(\rho) = F$$

$$0 = F + \upsilon * \nabla(\nabla(U_x)) \qquad (3)$$

Solving the differential equation for

$$U_x$$

(mean velocity) above gives us:

$$U_x = \frac{F}{2 * \upsilon} * (x(x-L)) \text{Where L is the length of the tube in Lattice sites}$$
$$(4)$$

```
void moveParticles(){
for(int x = 0;x<xdim;x++){
        for(int y = 0;y<ydim;y++){
                //additional code for flipping some
                int flip_parts = particle_flip_w*
                        ((double)rand()/RAND_MAX)*n
                n[x][y][3] += flip_parts;
                n[x][y][5] -= flip_parts;
                flip_parts = particle_flip_w*
                        ((double)rand()/(double)RAN
                n[x][y][6] += flip_parts;
                n[x][y][8] -= flip_parts;
                flip_parts = particle_flip_w*
                        ((double)rand()/(double)RAN
                n[x][y][0] += flip_parts;
                n[x][y][2] -= flip_parts;
                }
        }
}
```
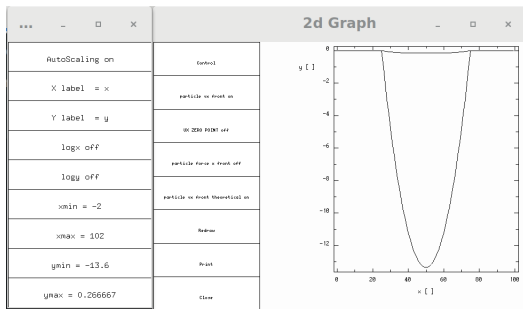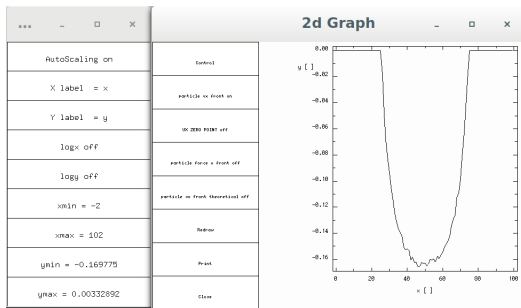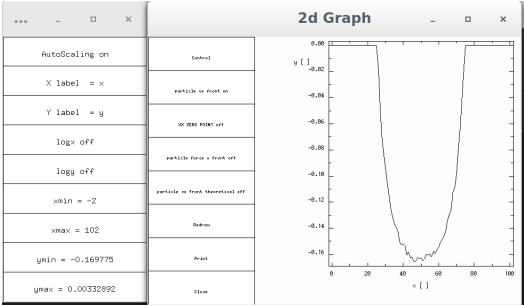
Results

Figure 10:

¡text¿

Results

Figure 11:

¡text¿

Results

Figure 12:

¡text¿

Conclusions and Final thoughts

- significant leakage for most walls
- partially working
- problem depth and complexity
- Approach 3 Issue might be solvable