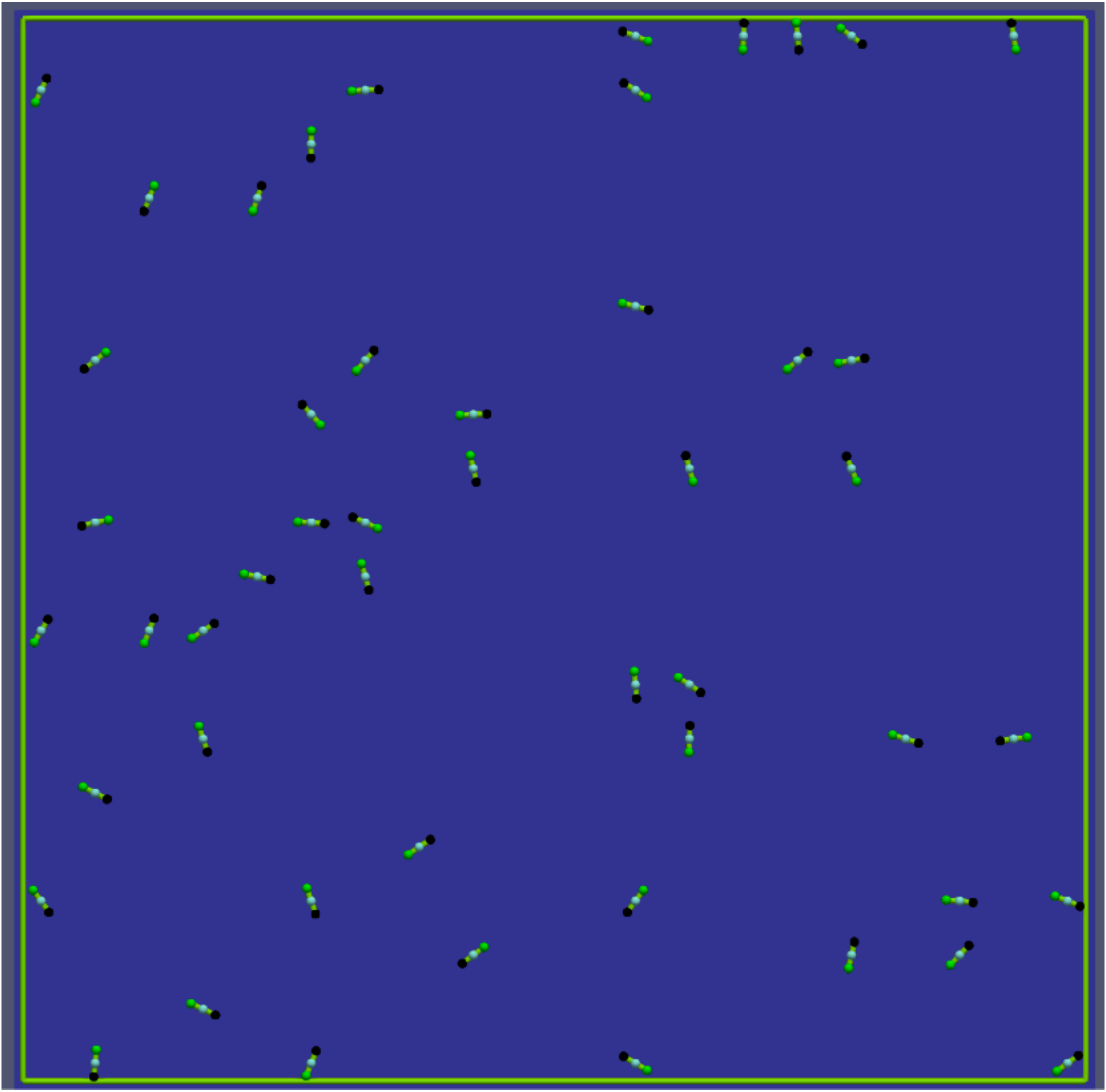
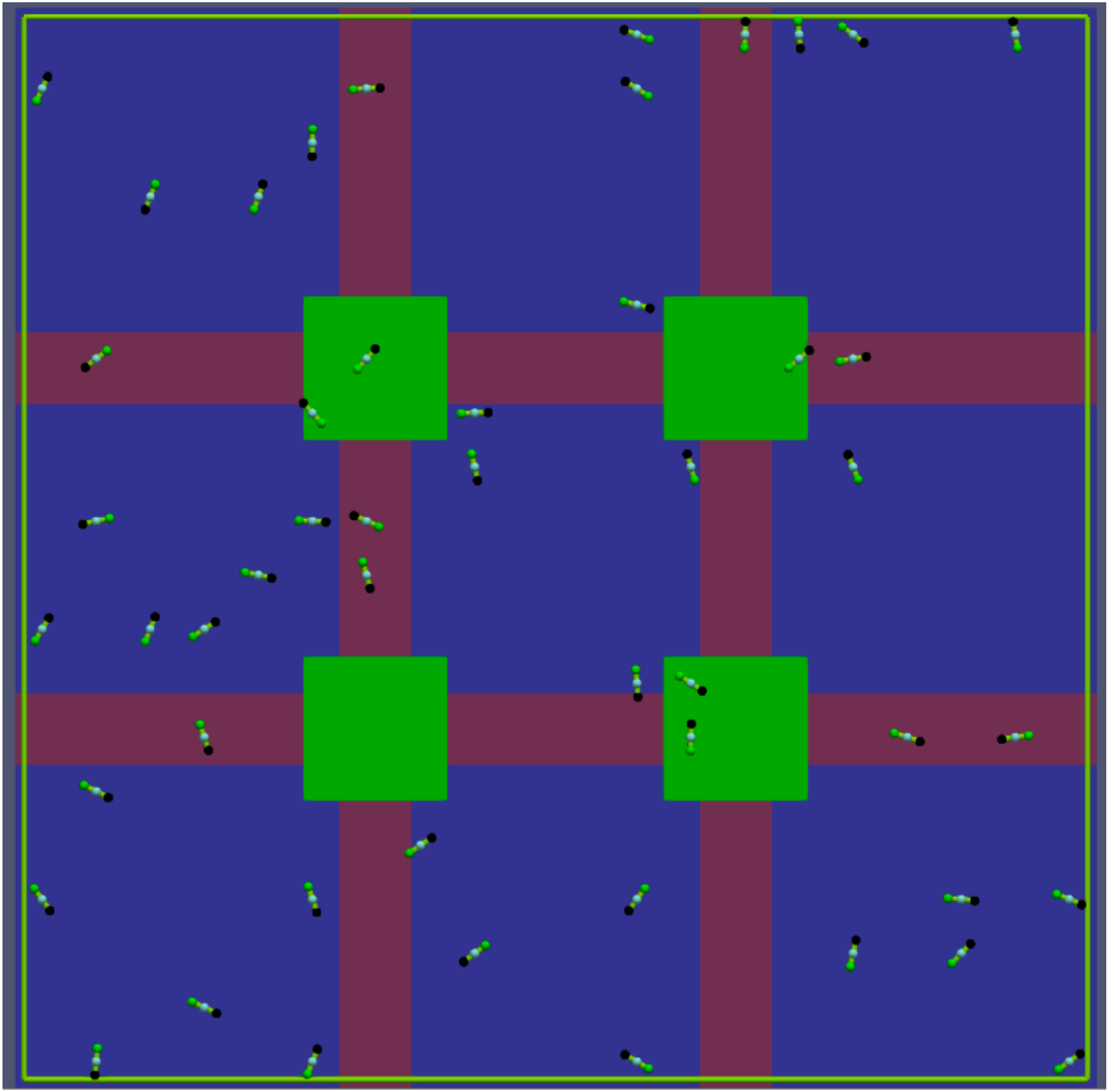
20

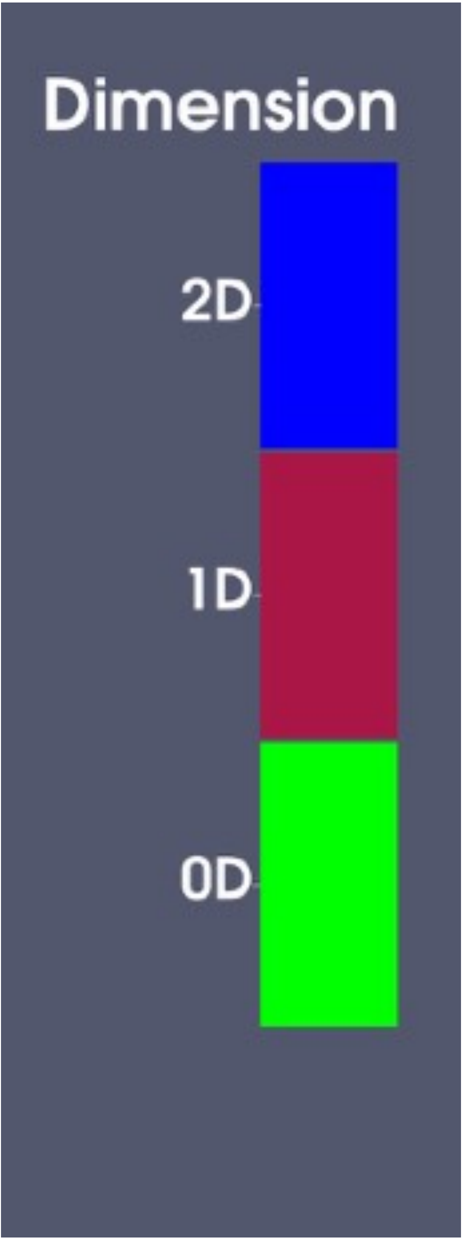




Initialize the Simulation Space

Figure 14: Initialized simulation space, with no cell complex expansion.

Figure 15: Initialized simulation space, with expanded cell complex, expanded cells are larger enough to hold matches.



Dimension
2D
1D
0D

Note: in the case of no subdivision the new approximate algorithm collapses into the exact.

Note: Any other form of subdivision is the approximate algorithm.

Initialize the Simulation Space

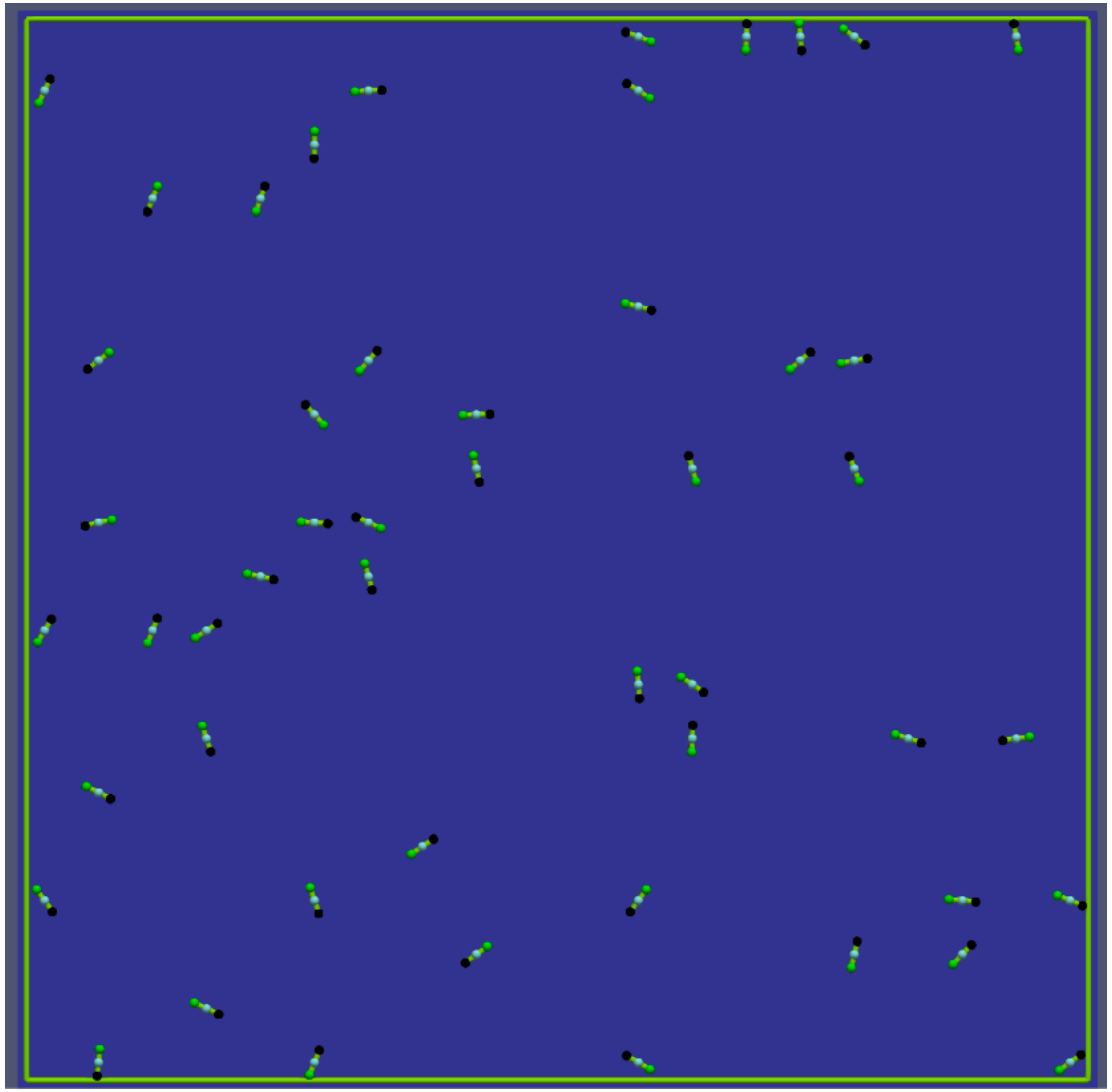
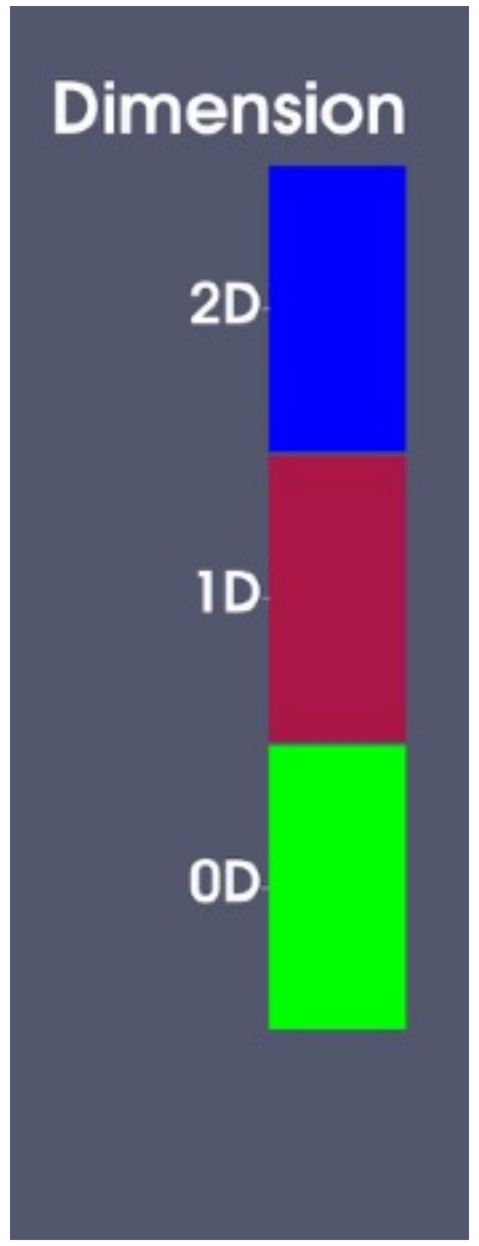


Figure 14: Initialized simulation space, with no cell complex expansion.



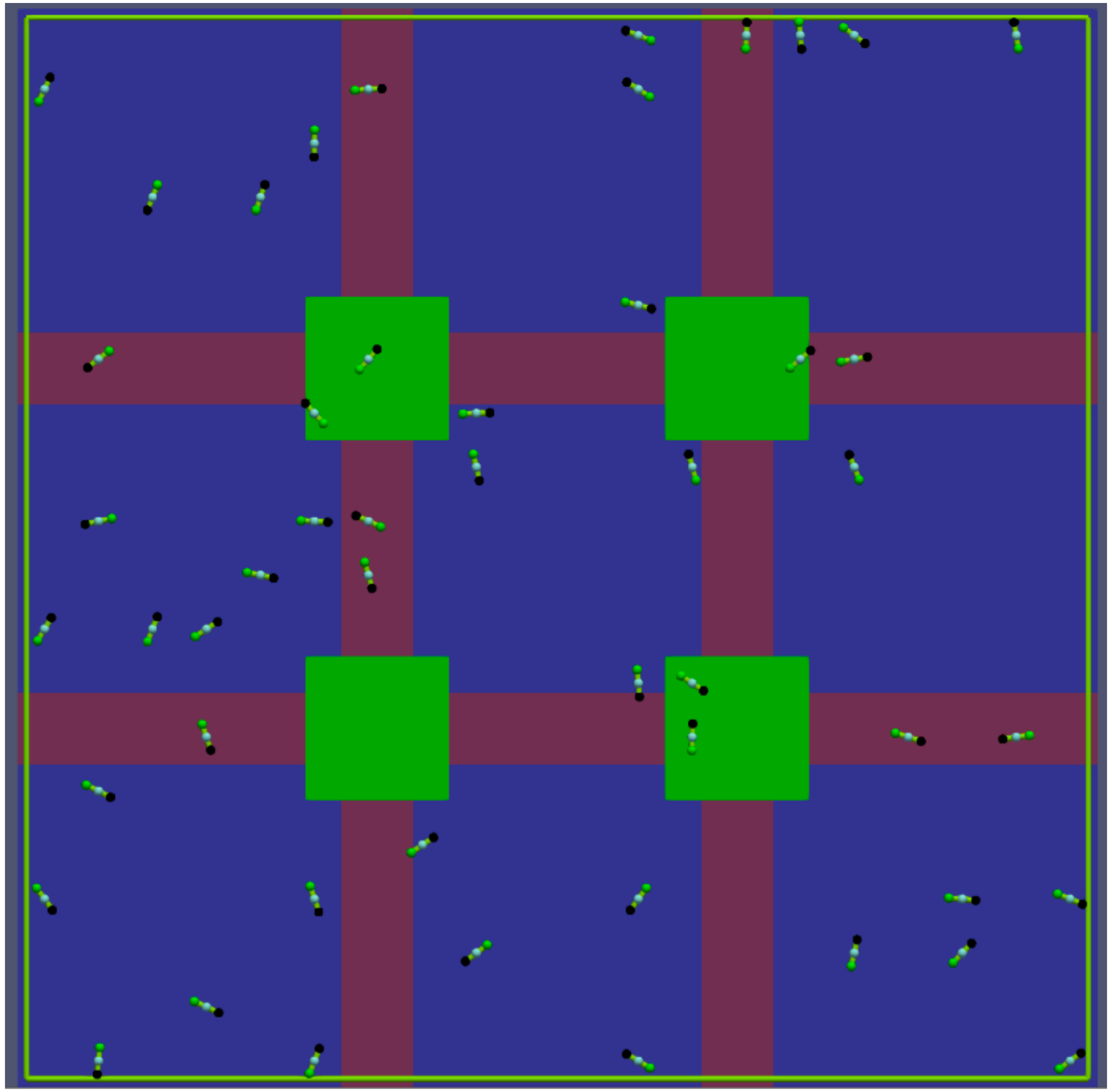


Figure 15: Initialized simulation space, with expanded cell complex, expanded cells are larger enough to hold matches.

Deriving the Approximate Algorithm

Operator splitting and approximation.

- A method to approximate e^{tW} is a first-order operator splitting algorithm that imposes a domain decomposition by means of an expanded cell complex.

- The decomposition corresponds to summing operators:

$$W = \sum_{(d)} W_{(d)} = \sum_{(d,c)} W_{(c,d)}$$

- Operators are summed over pre-expansion dimensions *d*, and cells *c* of each dimension:

$$e^{tW} \approx \left(\prod_{d\downarrow} e^{\frac{t}{n} W_{(d)}} \right)^{n \to \infty}$$

$$e^{t' W_{(d)}} = \prod_{c \subset d} e^{t' W_{(c,d)}} \quad \text{where} \quad [W_{(c,d)}, W_{(c',d)}] \approx 0 \quad \text{and} \quad t' \equiv \frac{t}{n}$$

$$W_{(c,d)} = \sum_r W_{r,c} \equiv \sum_r \sum_{\left\{ \substack{R \mid \varphi(R) = c, \\ R \text{ instantiates } r} \right\}} W_r(\, R \mid c, d)$$

- Note: $d \downarrow$ means we multiply from right to left in order of highest dimension to lowest.