Experiment:

Algorithm 4 was implemented in MATLAB using square block robots with unity width. The simulation was run for four swarm sizes, n= [8, 46, 130, and 862]. The swarm sizes are determined by the pattern input given to the code. This pattern is the required final configuration of the robots we want in the build zone (ref to fig.5). Once a pattern is given a tight bounded map is drawn following the scheme in fig 5 and robots are initiated in the staging zone.

Two plots illustrate the algorithm’s performance. Fig 4 shows the number of robots in swarm ’n’ vs the total commanded moves (in green). Observation of this graph gives us a relationship between the swarm size and the total distance the algorithm commands all the robots to move. The relationship is approximated by the fit line with equation . **In the first graph(fig 4) we take ‘ε’= 1 and implement the algorithm for all patterns**

y = - 0.017\*x^{2} + 240\*x - 4700

Fig 4

**The green line is the total commanded moves given to all the robots by the algorithm.**

**We observed that the value of ‘ε’, greatly affects the total number of moves of commanded moves on the robots. The smaller the value of ‘ε’ the greater is the number of moves required to move a said distance. The second graph plot(fig roboplot) is that of ‘ε’ vs ‘total commanded moves’ which is the number of moves the global controller commands onto all the robots. Here we take the pattern**

**‘ROBO’ and calculate the ‘total commanded moves’ for ‘ε’=[ 0.25,0.375,0.5,0.625,0.75,0.875,1]. We can notice an exponential fall in the total command moves. It can be inferred that there is a tradeoff between area of bounded region and the number of moves. For smaller ‘ε’, smaller is the required area to implement algorithm but the longer is the time taken to complete the algorithm. Fig robo**

**Unit of ‘ε’ is the ratio of minimum clearance value to the body length of a robot.**