Path Planning in Static and Dynamic Environments Using Genetic Algorithm

Hadji Isma

INTRODUCTION

One of the most important problems in the field of mobile robots, is the problem of path planning. The problem of path planning may be addressed in several types of environments. It could be in free space where the only problem is finding the shortest path, or in an environment with static or dynamic obstacles among which the robot has to navigate.

This project proposes an effective approach for path planning in an environment with static and dynamic obstacles using Genetic Algorithm. The algorithm will not only find the shortest path but also the smoothest possible path while avoiding any obstacles that the robot might encounter.

Two different flavors of the algorithm have been developed and compared. The first one plans the whole path at once, this has been applied to static environments. On the other hand the second one does a step by step path planning and this version was applied to dynamic environments. The pros and cons of each approach are discussed.

METHODS

Environment representation

First thing that was considered in order to implement a successful path planning is the representation of the environment. The method used is composite space map in which the whole environment is decomposed into small cells making our environment look as a grid on top of which the robot has to navigate as shown in figure 1. In this representation, each cell in which an obstacle falls is marked as occupied and the rest is marked as free. For simulation purposes, an environment of 2000×2000 cm was decomposed into cells of 50×50 cm.

The main challenge when using Genetic Algorithm together with this cell decomposition approach is finding a trade of between speed of convergence and accuracy. Decomposing the space as we did ensures greater accuracy but also imposes a very big search space that can be computationally expensive. Therefore, in order to reduce the time of convergence, a constraint was imposed on the choice of the initial path. This constraint consists in forcing the initial paths to be made of directly adjacent cells.

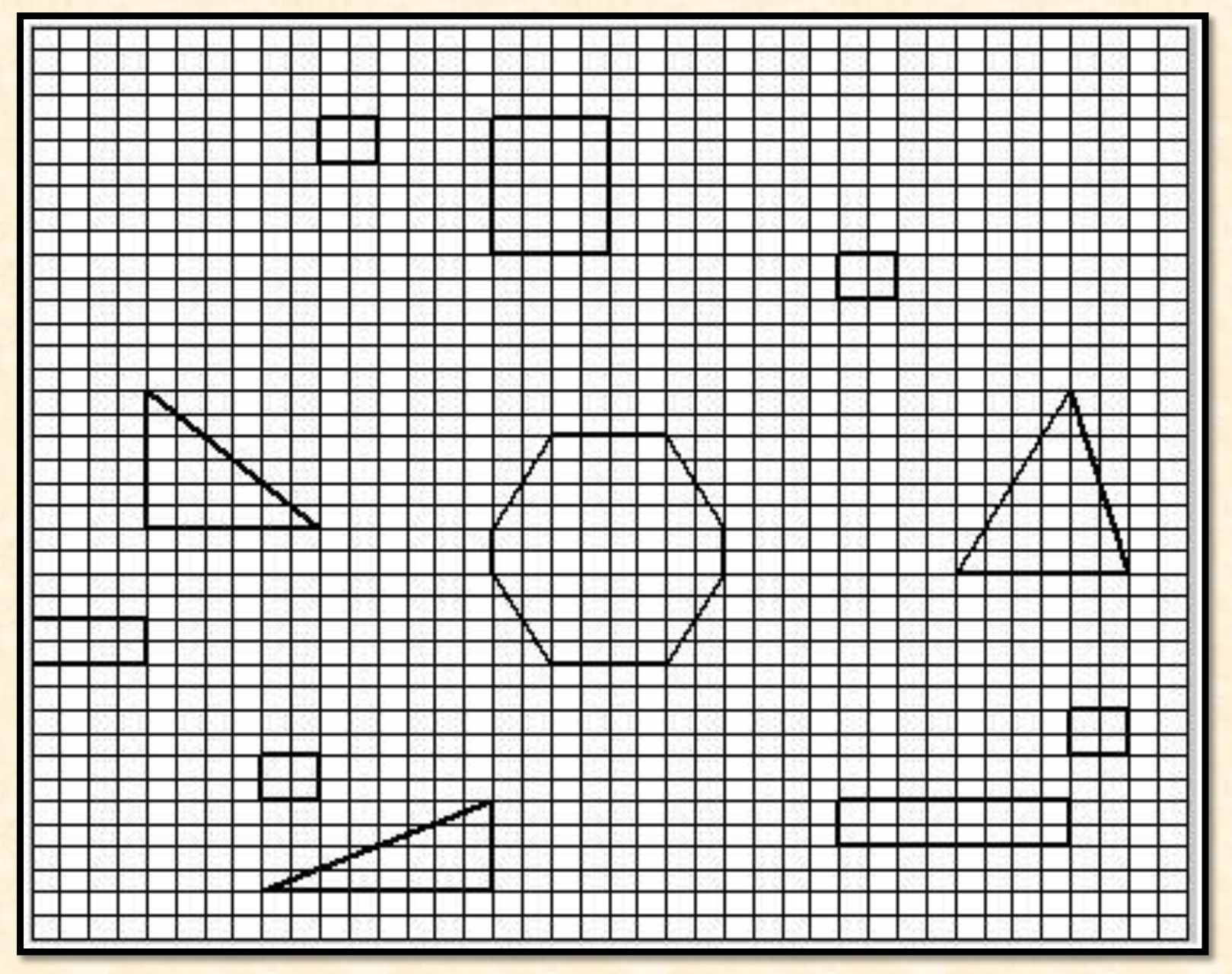

Figure 1. composite space map

METHODS

The evolutionary path planning algorithm

1. Initialization

The algorithm starts by randomly choosing an initial population of 50 different chromosomes. Each chromosome being a two dimensional vector made up of all the XY coordinates through which the path is going to pass in addition to the starting and ending point that might be selected by the user. As previously stated, in order to reduce the search space only three neighboring movements are allowed when transitioning from one gene to another.

2. Fitness function

The optimization algorithm takes into consideration three different constraints. First constraint is avoiding obstacles. This is solved for when we create a map of the environment, where we assign a penalty β to each cell containing an obstacle. Second constraint is of course minimizing the traveled distance. This is done by computing the Euclidean distance between each two genes.

Finally, the last constraint is the smoothness of the path. This is done by imposing a penalty α to paths with sharp turns. Sharp turns are determined by computing the inner product of each two neighboring points. It should be noted though, that the penalty for sharp turns is smaller than the one for collisions because the ultimate goal is traveling the shortest distance while avoiding obstacles, and according to the map it might be impossible to avoid an obstacle unless a sharp turn is made to avoid the collision.

The above conditions are put together in the form of the following fitness function: $\Rightarrow f(k)=d(k)+t(k)$

Where d(k) is the distance computed using the following equation $\Rightarrow d(k)=\sum \|p(i)\ p(i+1)\|+\beta(i)$

$\beta(i)=$ w1 if there is a collision in the path

0 otherwise

and t(k) is the number of sharp turns $\Rightarrow t(k)=\sum\alpha(i)$

$\alpha(i)=$ w2 if a sharp turn is made

0 otherwise

Evolutionary process

The proposed algorithm uses one point crossover operator that randomly takes two parents and produces two offsprings and one mutation operator that randomly picks points from a chromosome and replaces them with another point from the search space.

The selection of the parents that will be used to produce offsprings is based on elitism, where 1/2 of the fittest parents are chosen with a probability of 50% to produce offsprings . Then, the mutation operator is applied on the offsprings with a probability of 50% to mutate. This probability of mutation is chosen to ensure wide exploration of the search space. Finally, the parents and the offsprings are set to compete for survival to next generation. This whole process is repeated until the fitness falls below a certain threshold.

The proposed algorithm works well for moving in a static environment, however to handle dynamic obstacles the same algorithm was applied iteratively to both learn the environment and build the path in a step by step fashion. To accommodate for that, variable size chromosomes where used to build the path as small pieces put together to form the whole path.

RESULTS

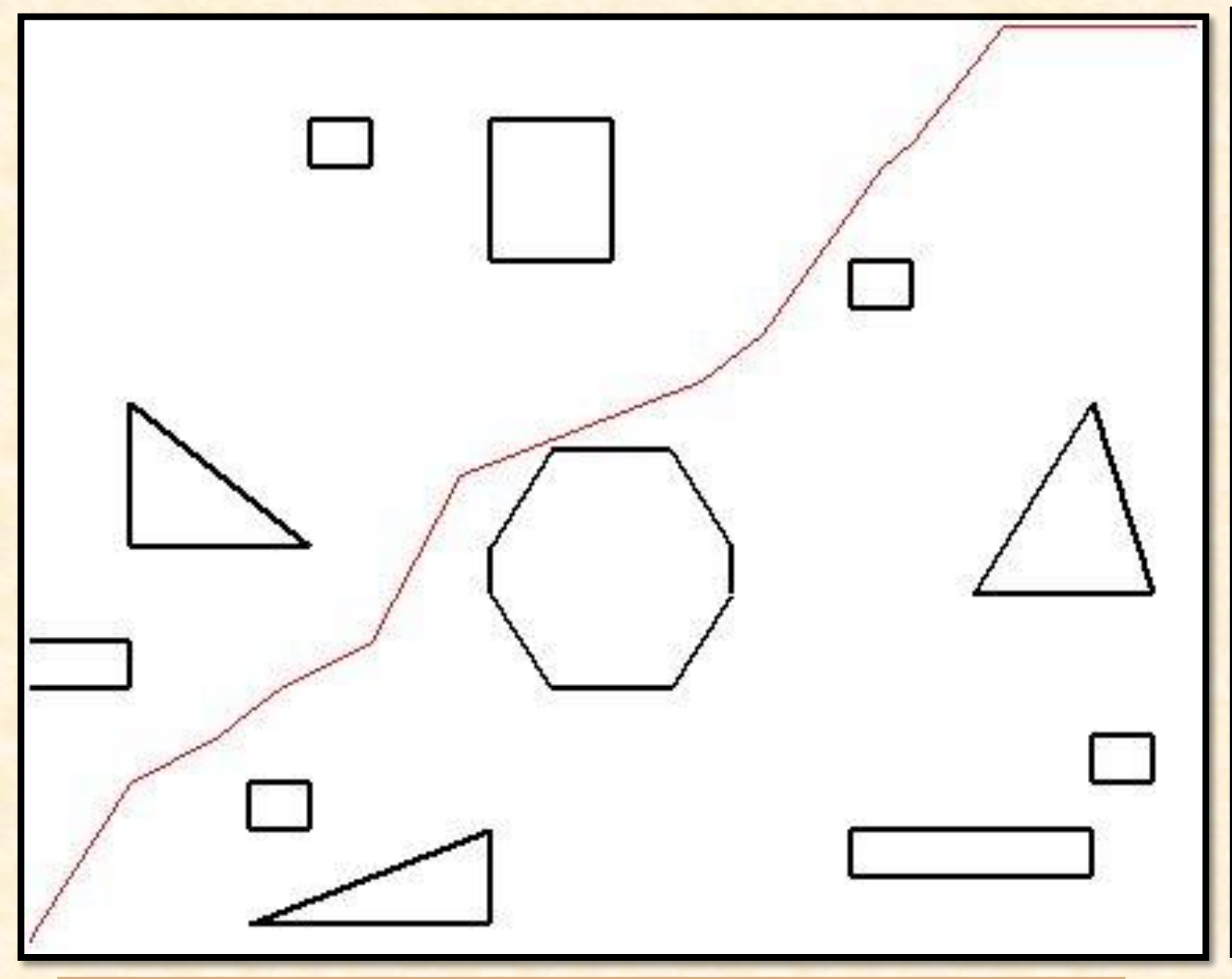

Figure 2. static environment Map 1

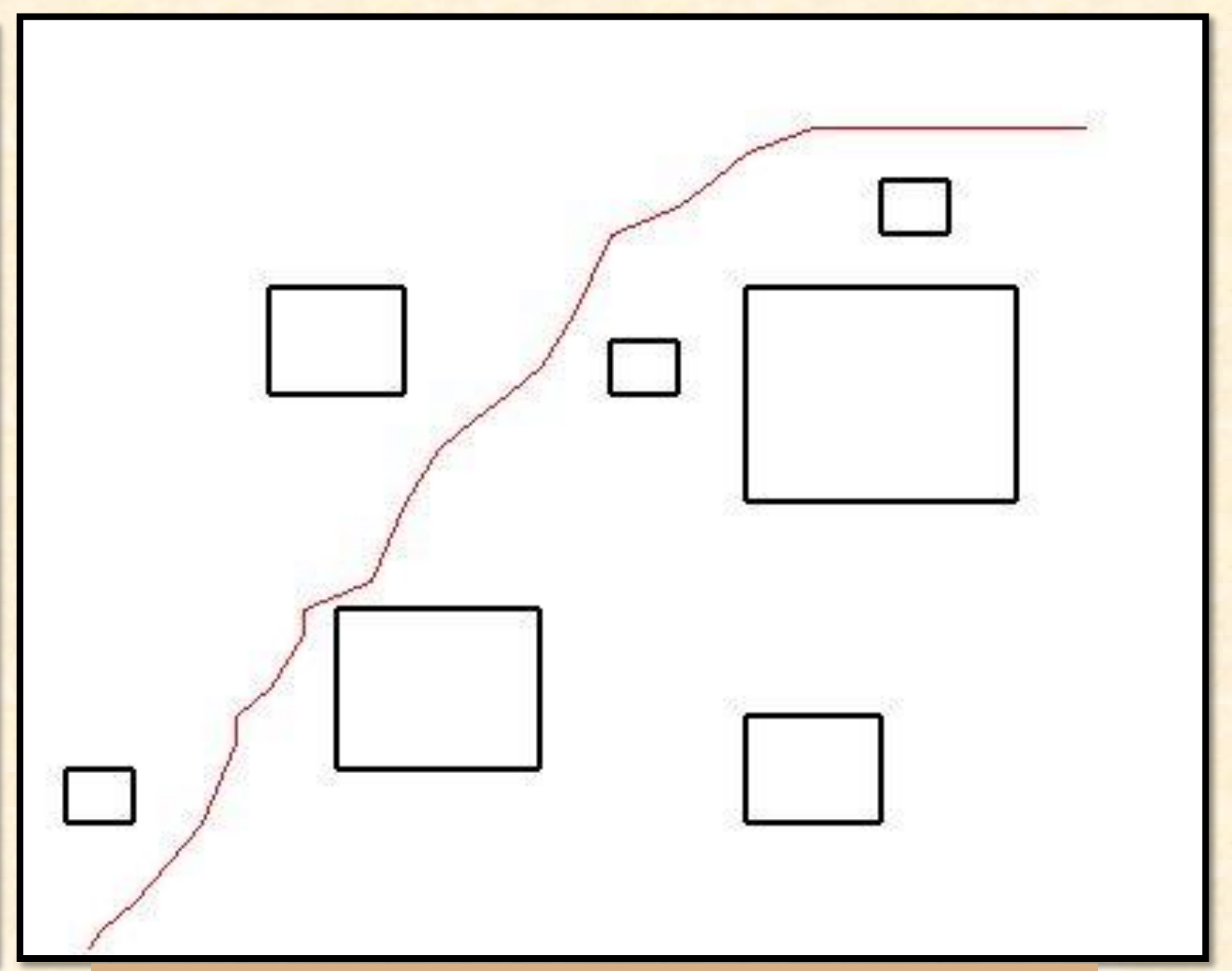

Figure 3. static environment Map 2

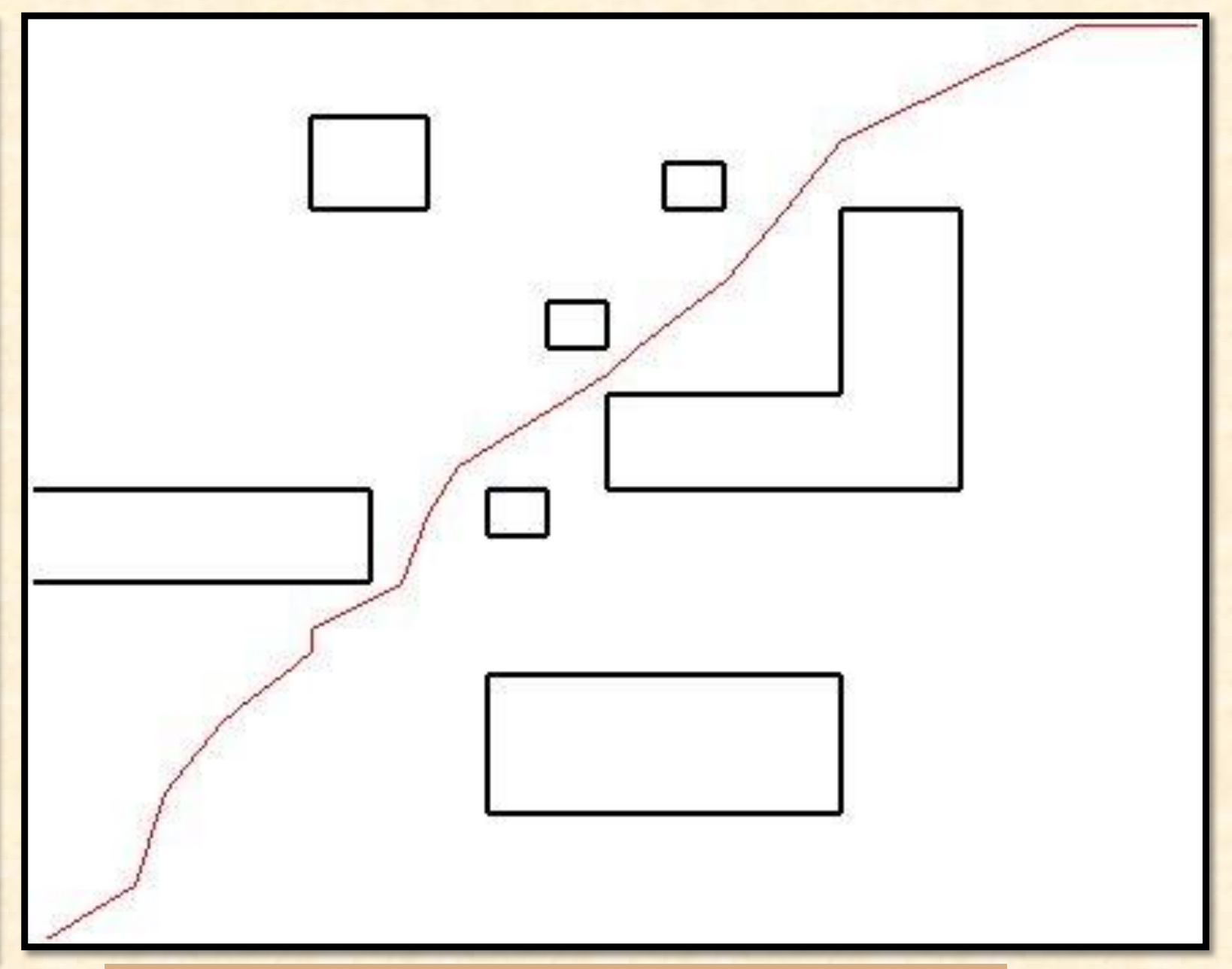

Figure 4. static environment Map 3

In this case the environment is preset before hand and therefore the algorithm has a previous knowledge of all the obstacles it will encounter.

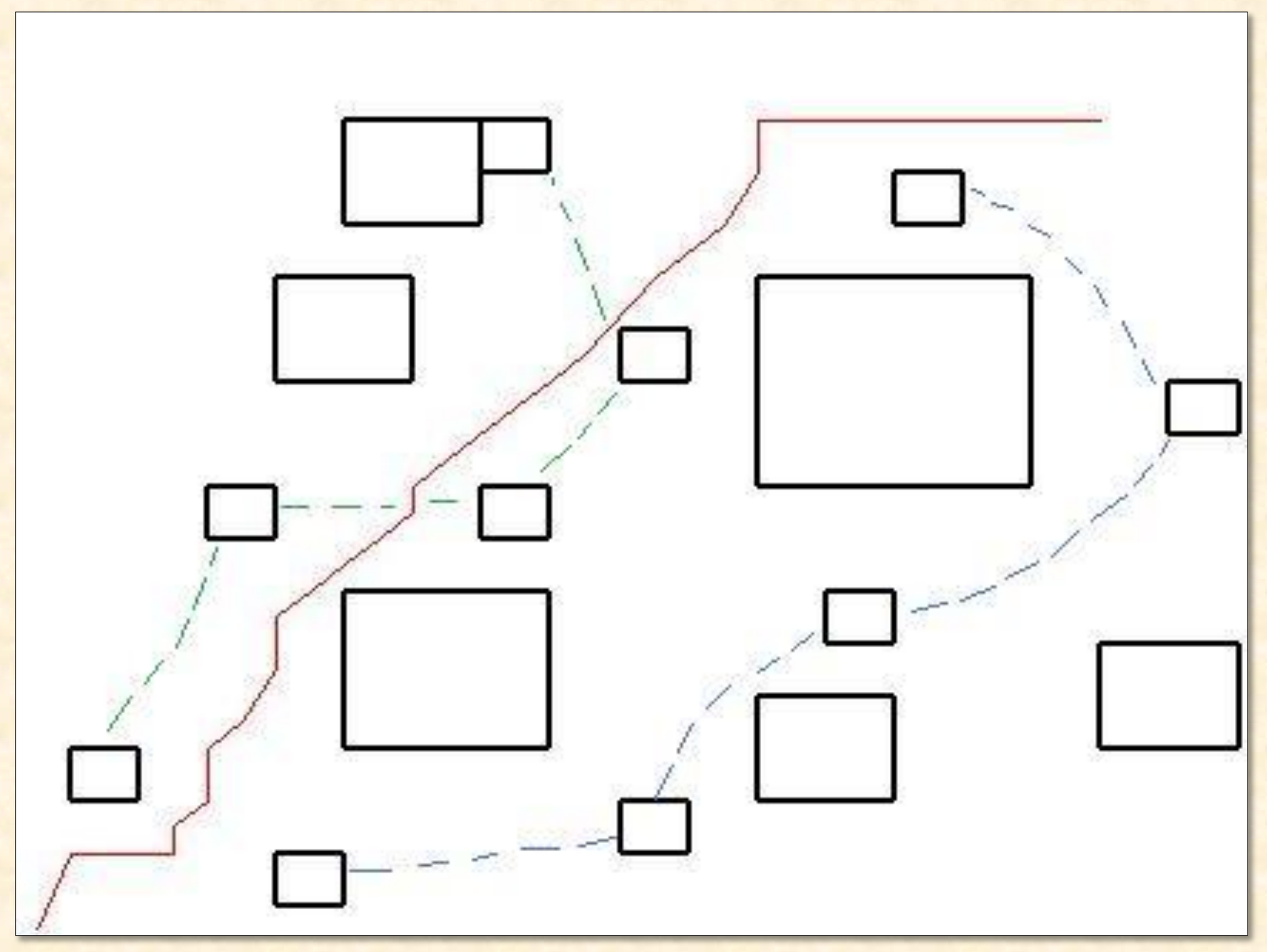

Figure 5. Dynamic environment Map 1

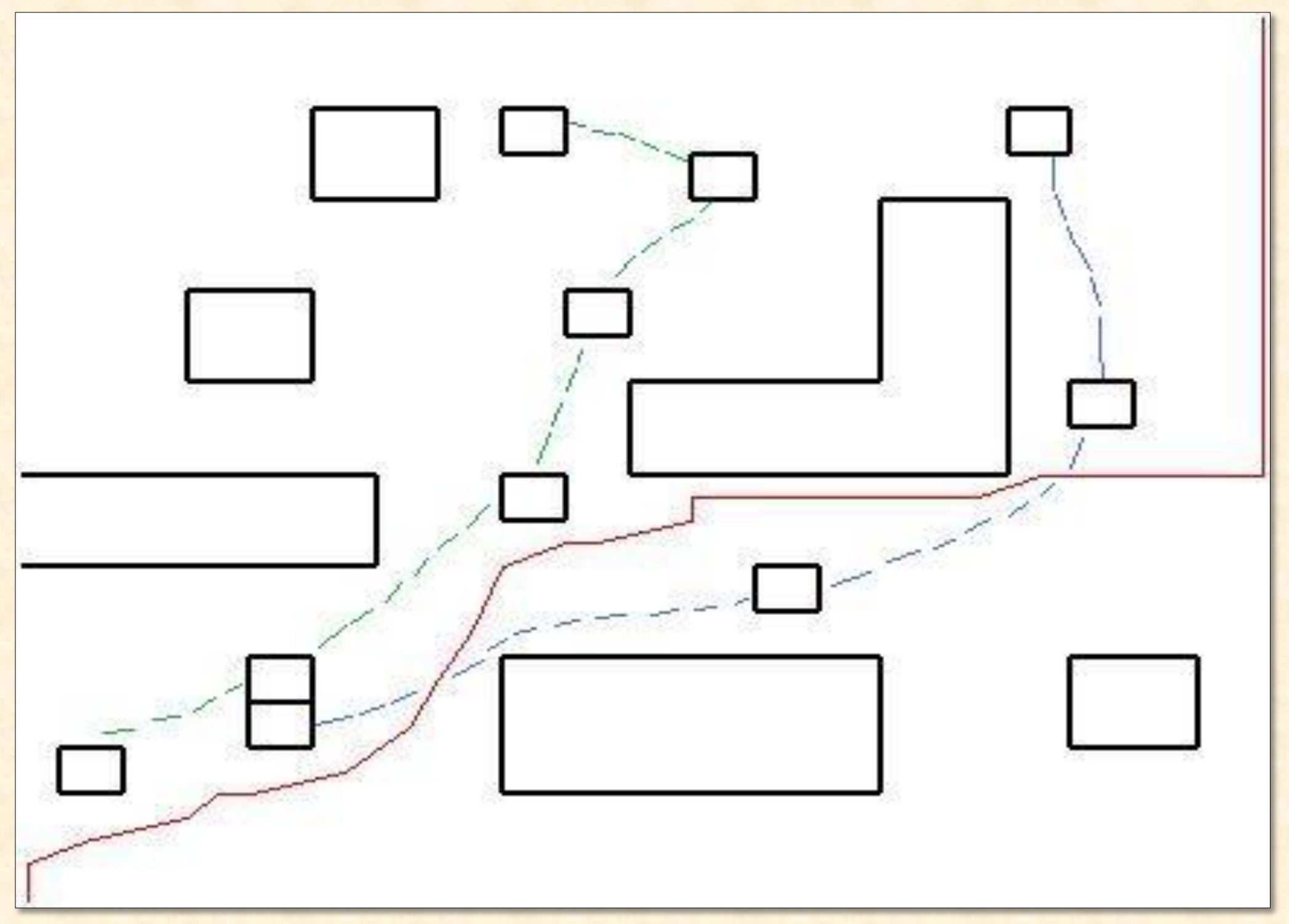

Figure 6. dynamic environment map 2

In this case the obstacles appear randomly on the fly while the path is being computed. Therefore the path had to addapt according to the obstacles that appear on its way.

CONCLUSION

The proposed algorithm was tested under different environments and proved to be very effective as it succeeded to find near optimal solutions every time. The algorithm for navigating in a static environment produced a solution in an average of 86 generations while the algorithm for dynamic environments produced a path in approximately twice as many generations. This last result being quite logical given that the second flavor of the algorithm runs iteratively and has to account for more computational complexities, such as a variable length chromosome.

REFERENCES

[1] Yanrong Hu, Simon X. Yang, Li-Zhong Xu and Max Q.-H. Meng, "A Knowledge Based Genetic Algorithm for Path Planning in Unstructured Mobile Robot Environments," In proc. Of IEEE International Conference on Robotics and Biomimetics, Shenyang, China, pp. 767-772, 2004.

[2] H. Mahjoubi, F. Bahramin, and C. Lucas, "Path Planning in an Environment with Static and Dynamic Obstacles Using Genetic Algorithm: A Simplified Search Space Approach," In *Proc. of IEEE Cong. Evolutionary Computation*, Vancouver , Canada, pp. 2483-2488, 2006.

[3] Ismail AL-Taharwa, Alaa Sheta and Mohammed Al-Weshah, "A Mobile Robot Path Planning Using Genetic Algorithm in Static Environment," Journal of Computer Science 4 (4): pp.341-344, 2008.