Geometric A* implementation on multi-robot environment

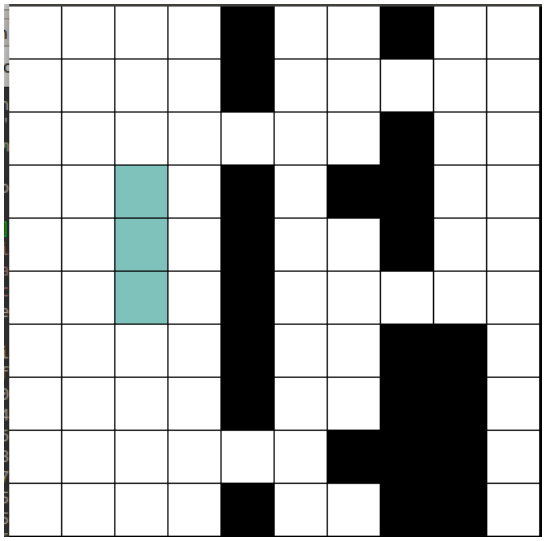

Fig.1 – The work environment with the three robots represented as light blue boxes and the obstacles in black.

Abstract

In this report I will show the application of geometric A algorithm for pathfinding implemented on a multi-robot system. First I will show the A* algorithm apply to the agents in a grid world and the problems that can arise in this context. Then i implement this method with two improvements that solved most of the problems : using the visibility graphs and the post smoothing. Finally I will show the obtained results of this methods.*

Summary

The report is organized so :

Introduction

In this work we use the geometric A* algorithm for find a path, or well say a sequence of states, from initial to final position for a team of omnidirectional robots (in this case three) trough a bidimensionale environment covered by casual obstacles. In the state of the art are present many works concerned the utilization of this algorithm to find a path for a single agent; there are also present many variants and typologies of A* each of them suitable and designed for specific purpose. In this work I used the main principles of this algorithm to adapt it to multi-robot environment : this domain of study is living a great wave of interest from different areas of research, from medicine to rescue and emergency, from military industry to nanothecnologies application and to other studies fields. The key of the success of this multi-agents sistems rely on the robustness of the system to failure, cheapness of mechanical parts ansd the capacity of many agents to perform a single shared task. One important field of work with this agents is for navigation purpose : to achieve a location trough an environment full of obstacles, once the agents has the knowledge of the environment, they must plan a route to achieve a final destination submitting to particular constraint different situation by situation. The background situation for this work and in which I tested planning algorithms, is the navigation of three robot trough an environment composed by free locations and obstacles locations : the three agents for navigate on this environment must respect distance constraint among them and so this condition forces them to take right actions in order to respect this condition. Then in this report, step by step, I will explain the adopted implementation, the problems arisen from this choice and possible solution.

Geometric A*

The A* algorithm is an algorithm working on dependent-domain heuristic breadth-first search : the heuristic is used to direct the direction of the search in the graph search to the goal node.
The geometric A* is its variant applied to the geometric world : to adapt it, we must do a series of simplification . First, original A* work on graph while geometrical on 2D or 3D environment : we can discretize it using a grid and each small square will be like original node. Second, the heuristic will change : initially it is based on a some domain-dependent function, while here it takes in consideration the metric distance information between position on the grid. Surely this simplification bring into the system a kind of resolution error that will be treated later. So the first step in our work is importing geometric A* in this world and adapt it to a multi-agents set up. We do this changing the state dimension : instead of using a state of 2 dimension (x-y position on the grid) we use an extended state of dimension 2 x n (so in our case with three robot we have a vector composed by six elements). So in order to compute the change of position of the agents during the computation, we find all possible moves that the agents can do (at each step move one, two or all of them).
After it entries in the main loop of the algorithm and it will run until we reach the final position or the length of the fringe is equal to zero.
It takes the node with small f as current node and, if the currend node is different from the end node, for this node generate all its children.
In this part we concentrate the core of the multi-agent part : the generation of the childrens is constrained to problem specific. In particular the constrain set the distance which every robot has to respect to other robots; if the generated node doesn't respect this condition is discarded.
In this part also the new position which are outside the grid or intersect an obstacle are discarded. Then we loop trough the children for computing the f function :

F = G + H

The g function is simply computed ading to the current node's f function a small constant, while the heuristic function is the geometric distance from the node to the final position.

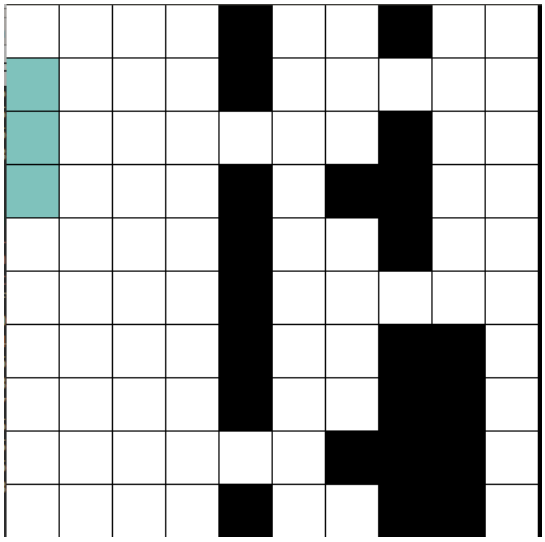

Fig.2 - Over the three robots are showed in the initial position. The grid is an insidious environment where the robots cannot pass trough maintening same initial position, but must adapt to it.

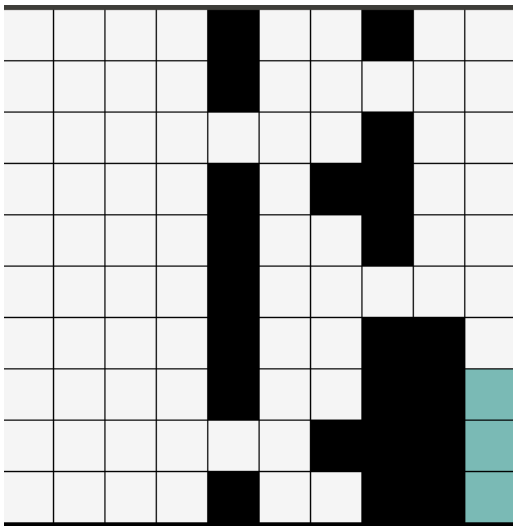

Fig.3 - Over it is showed an instantaneous of the environment. The three agents represented by three blue boxes reached the final position at the left-down

side at the grid passing trough insidious narrow passeges.

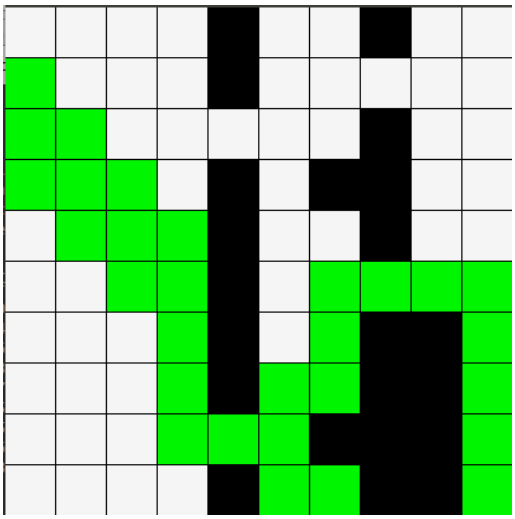

Fig.4 - Instead in thus image is showed the path executed by the team from the initial to final position.

So the A* algorithm is an optimal strategy to find the best path between two locations, but due to the approximation the minimal path on the grid is not the real minimal path : in fact in the grid are allowed only movements following the squares and diagonals and this lines cannot be inside the occlured areas, which also these are approximated to adapt them to grid areas. The allowed moves are a restriction (in the reality the agents can move following n-polynomial trajectory or other continue function they can do) and this restriction lead to the fact that the only goal we can reach is the sub-optimality. Increasing the resolution increasing a more realistic representation on the obstacles, but it doesn't change the fact that the grid constraint the movements.
So the real path may become impossible to find due to this simplification. However in this project I' ve implemented two improvement from the original geometric A* which improve the performance of the system : the Visibility graph and the Post-smoothing.

I improvement : Visibility Graph

The visibility graph is a structure created taking in consideration only the occlured areas and the agents. To avoid to be constrained in a grid structure, this strategy rely on the line-of-sight strategy : considering only previous objects and assuming that obstacles have a square form, we track a line from each corner present in the scene among them. In this way I create a fully connected graph between each objects' corner. This strategy rely on the well known idea that the shorte path between two point is a line.
Then I calculate for each of this line the length and finally I computed the shorter path using the previous A* algorithm but this time based on this new graph. The path that we found can be followed step by step by a single agent, but to adapt this situation to many agents it is used as an heuristic : in the A* algorithm choosing new children, the function will choose whose that will be closer to the path on the visibility graph. In the code, first we create the visibility graph, compute distances between points and using A* for a single agent once to find best path. Once it returned the best path as a sequence of points, we use this path as an heuristic for the equipe of robots, mixed naturally with the heuristic which approach to end position. So the system try to achieve two different goal respecting the multi-robot's system constraint : reach final destination following closest as possible shorter path precomputed. This typology of variant maybe is not the ideal for online application since for more complex environment maybe more computationally expensive.
Below I show three sketch from the simulation:

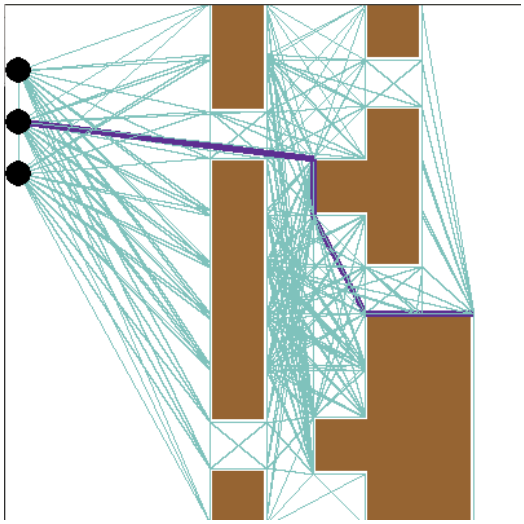

Fig.5 - In the figure are showed the three robot in black at their initial position,the whole visibility graph in light blue and the final optimal path in blue.

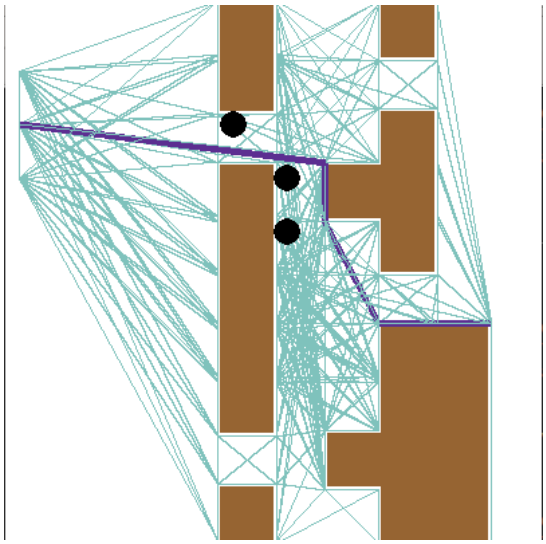

Fig.6 - In this other figure we can see the ability of the three agents to pass trough a narrow passage following the reference path.

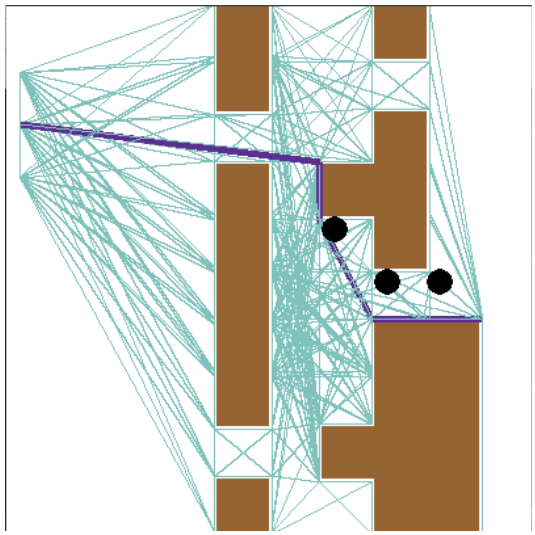

Fig.7 - In this final shot the three robot reach its final destination. This point is setted for only on agents, then the other two must adapt their behaviour and their position to achieve this point.

II Improvement : Post Smoothing

Another possible improvement to improve geometric A* performances is the post-smoothing. It is based on the idea to smooth the original path using the line-of-sight property . Initially it take as input the original optimal path from A* in the grid and then it try to optimize it. For each singular point in the original path, starting from initial position, it try to find the maximum segment from that point to another point which belong to the path : at the end, when it controlled all points, it choose that in order to obtain the longer segment and discard from the computation all intermedia points. This iteration is executed until it reach the final point. The system is simply based on the concept to improve the approximation introduced by the grid assumption. Also in this case, like in the visibility graph context, the new path is taken as reference fot the heuristic to compute the global state sequence.

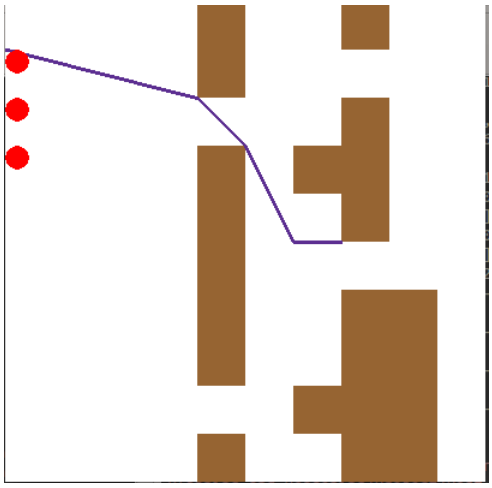

Fig.8 - In the image over are represented the three agents at the initial position.

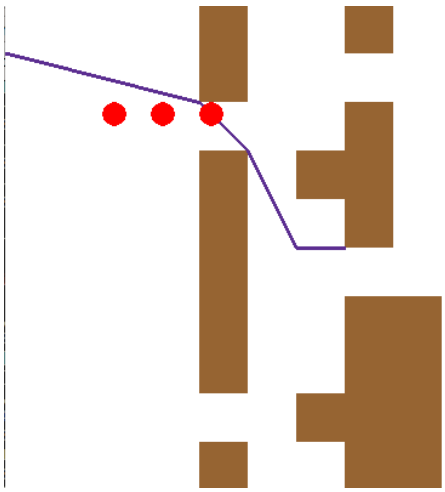

Fig.9 - Like the previous case, the team of robot achieve the task of passing through narrow passages.

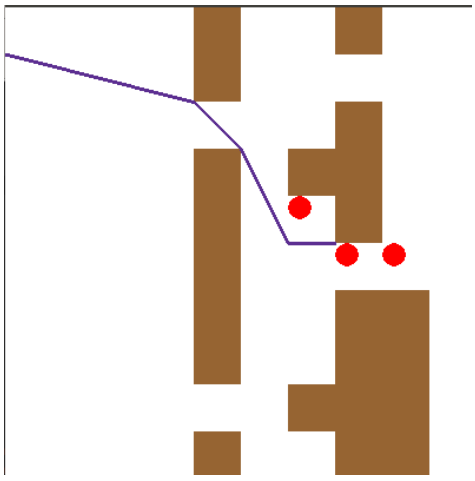

Fig.10 - In this last image the goal is reached following best path.

Conclusion

In the end we can say that the geometric A* algorithm is an optimal tool to use in the multi-robot domain. There are many variants of this typology of search strategy and this makes this algorithm very versatile for many purpose : like seen before, variants can be applied to increase its performances. Over visibility graph and post-smoothing there are other possible variants suitable for this aim, like Field* or Theta *. Finally the results obtained with this work are promising and surely it can be adapted to more complex situation concerned maybe a more number of agents, a 3D environment for UAV tasks or maybe also insert in the heuristic of the algorithm different features like different job for different agents or composition among them purposes.