Localisation in a Known Environment

p15194468

Sam Fuller

2019

# Implementation

## Parallelisation

Parallelisation allows the program to simulate more particles, the more particles used the faster the robot location can be discovered. In order to achieve parallelisation, the program had to be split into a series of functions that could be mapped onto the list of current particles. Testing on my personal computer parallelisation allows 8,000 particles to be simulated if they are displayed on the graph, or 16,000 if they are not displayed.

## Particle Definition

A particle is a theoretical point the robot could be; therefore, they contain x and y ordinates and a heading angle (in radians). In addition to the basic location information, the definition contains a certainty value, the certainty represents the chance that this particle is in the same location as the robot.

## Changing the particle limit

In order to make the program as accurate as possible, but also complete its action within one cycle the particle limit is changed. If the program takes more than 1 second to complete a cycle the particle limit is lowered, conversely if the program takes less than 1 second to complete a cycle the particle limit is raised.

## Updating a particle

### Repopulation

Initially random particles are generated; however, as particles are pruned the particles with the highest accuracy are used to help place the new particles. The new particles are placed in a normal distribution around all high accuracy particles, the sigma value of the normal distribution is defined by the certainty value of the high accuracy particle, the higher the certainty the lower the sigma value and the more likely it is to be placed close to the existing particle.

This means that an area with high certainty will become tightly concentrated, fine tuning to the position of the robot. However, an area of low certainty will become scattered allowing for a high chance of a particle existing in a similar location to the robot.

### Certainties

Each cycle the certainty of each particle is re-evaluated, the certainty is the mean average of the occupancy grid values of the sonar locations. The certainty re-evaluation consists of a parallel and serial function, the majority of the code is in the parallel function; however, the parts that can not be parallelised (or would take more time than running in series) are in the serial function.

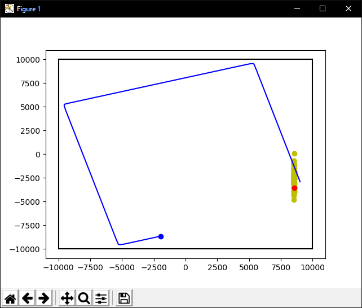
### Prune

Each cycle 2 methods of pruning are used:

1. All particles in invalid positions are removed (outside of the map, sonar readings that are outside of the map)
2. The worst 90% certainty particles are removed

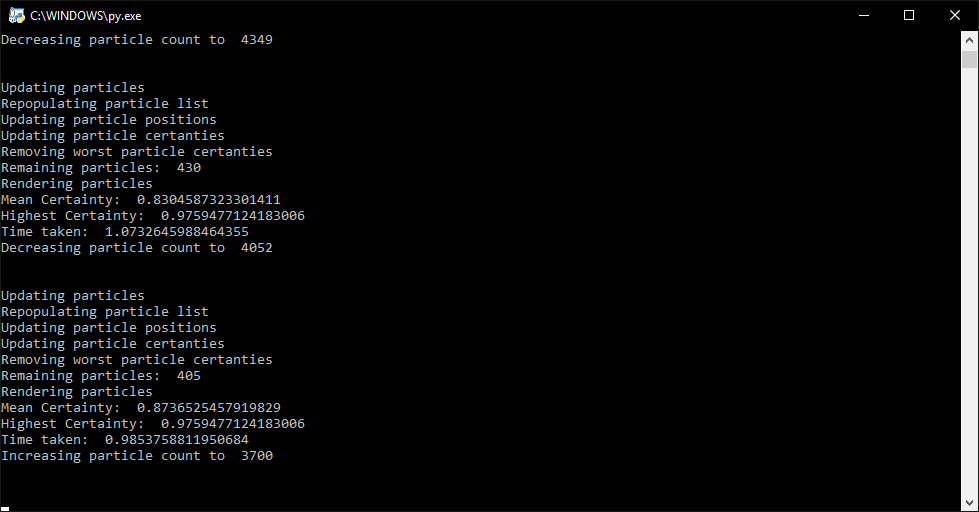
Removing the worst 90% of the particles allows for only the accurate particles to be kept, the large percentage allows for the program to quickly determine where the robot is.

## Display



The graph shows 4 things:

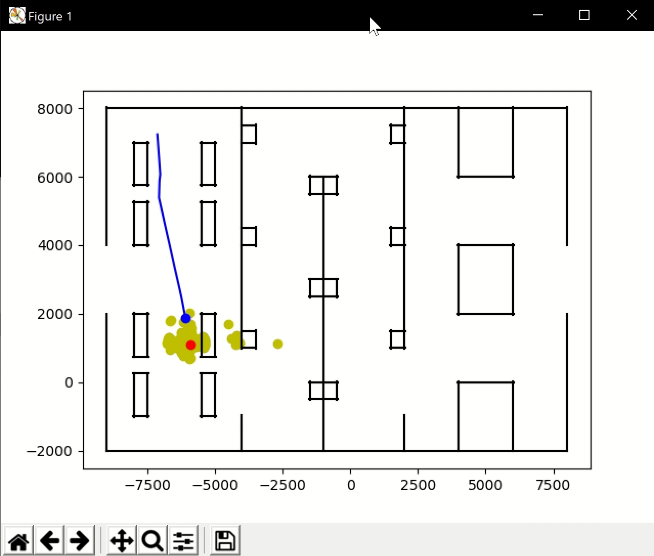
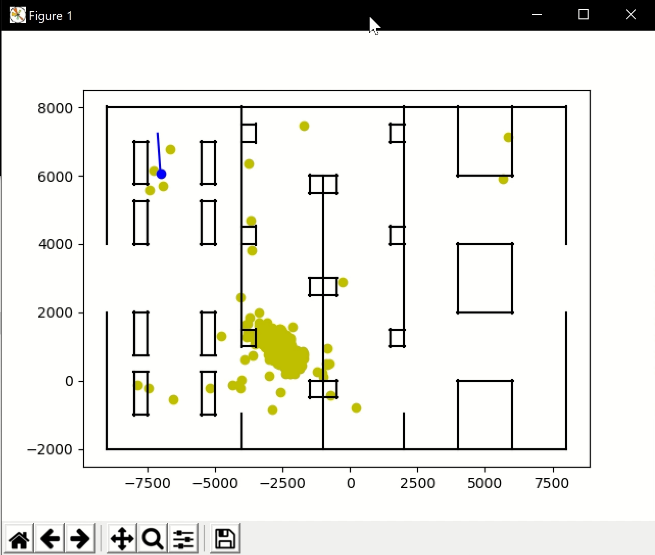
* Blue point: the robots current position
* Blue line: the robots path
* Yellow scatter: all particles
* Red point: the particle with the highest certainty.

The console shows several important values.

* The number of remaining particles
* The mean average certainty value of all current particles
* The highest certainty value of all current particles
* The time taken to complete the cycle
* The particle limit for the next cycle

# Testing

## Square

The program was tested on the square map, the program locates the robots position quickly. It is important to note that the square map has a rotational symmetry of 4; therefore even though in many places the robot and the particles are in different places, the particles are still in a correct place.  
  
  
  
  
  
  
  
The program was tested on the mine map where the robot took longer to locate its position, this is due to the closer spacing inside the map and more of the occupancy grid having non-zero values.