4D-Particle filter localization for a simulated UAV

Anna Chiara Bellini

annachiara.bellini@gmail.com

Abstract. Particle filters are a mathematical method that can be used to build a belief about the location of a robot in an environment that has a few known features, called landmarks. Each particle is a virtual copy of the robot, representing a possible robot's location, that moves in the environment as the robot does, and with an associated probability, representing the likelihood of each location. Through cycles of move robot-update particles, sense-get likelihood and resample, the probability distribution changes to reflect the current belief. The environment is divided in cells and the probability of being in each cell is the sum of the particles in that cell. The orientation is the weighted mean of the orientations of the particles in a cell. The demo uses the simulator built for the AIRobots EU project.

Keywords: Robot localization, Particle filters, Probabilistic localization, UAV

1 Introduction

Autonomous robots that move in an environment need to have a very precise knowledge about their own location in order to make correct decision about the next action or motion. While usually some information about the general location is available, it is most often the case that the error is too large to be sufficient for the robot, but the robot can use information from its sensors to refine this information up to the degree of precision that it needs.

For instance, a car on a road has a rough idea of its location thanks to a GPS and a map of the road, but the error of a normal car GPS is in the order of +/- 3-5 meters, which is not sufficient for a car to drive autonomously. Similarly, a robot inspecting a plant knows the map of the plant and might be initialized to a known starting location, but, as it proceeds around the plant, the accrued error gets too large to allow the robot to move in narrow corridors or pass doors. It may be the case that the initial conditions are unknown, making it impossible to rely on knowledge of the previous path to determine the location.

In each of these cases, there are distinguishable features that the robot can perceive, and whose location on the map is known, such as a tree that the car can see at the side of the road and that it can individuate on a satellite picture of the area, or a lamp on the plant's wall that is documented on the electrical project. Such distinguishable and known features are called landmarks, and are what the robot can use to refine its knowledge. However, the landmarks are often indistinguishable from one another, i.e. there can be several trees by the road, and probably all of the lamps in a plant are identical. Similarly, the information of the map could be incomplete, like a lamp could be off, or a tree might have been planted after the satellite picture was taken. Particle filters allow robots to localize themselves in these difficult situations.

The work described in this paper was done as an academic project within the AI Fundamentals course held by Prof. Paola Mello at the Faculty of Engineering in Bologna, using also material from an online course held by Prof. Sebastian Thrun at the Udacity website [4]. Techniques learnt in the two courses were applied in the simulation environment built for the AIRobots EU project [3], where the author has developed the 3D simulation part.

2 Particle filters

When information is so partial and uncertain and the environment grows in size and complexity, deterministic or analytical methods for localization are bound to fail, because the number of possible cases to be considered becomes too large just after a few percept-action cycles. Particle filters [1][2][5] represent the current belief of the robot as a discrete probability distribution over the entire environment, or at least over the portion of the environment that is being considered, such as the area individuated by GPS. In the middle of each percept-action cycle, after the perception and before deciding on the action, an update-resample step is taken and a new belief is built based only on the current belief state, the percept from the sensors (i.e. landmark sensing) and the previous action, so the computational needs don’t grow over time.

2.1 Multimodal belief

In an environment, there could be two locations A and B that are very similar with respect to the landmarks, and the best assumption that the robot can make is that it is either close to location A or to location B. This is true especially at the beginning of localization, when the robot has very little knowledge, so the probability distribution that represents the belief must be multimodal, i.e. have multiple possible peaks. Over the subsequent update-resample cycles this belief could become unimodal, but even when the best assumption is to be in one of a few locations, this information could be sufficient for the robot to take correct actions.

2.2 Particles

The way that this belief is represented is by making a large number of hypotheses about the possible location of the robot, and each of these hypotheses is called a particle. A particle acts as a virtual copy of the robot, and perceives and moves just as the robot does, but it does so on the map of the environment, not in the real one. The more the particles’ perceptions agree with the robot’s ones, the higher the probability that the particle is close to the robot’s location. After each update step, some particles survive or die according to their relative probability, but the total number of particles considered is fixed. The belief of the robot is given by the density of the particles: the more particles are in a specific area of the map, the higher the probability that the robot is in that area. Motion and perception on the map are computationally very easy, involving just simple arithmetical operations like addition, multiplication and roots, allowing the use of very large number of particles and precise localization.

Figure 1 shows a sample 2D world where a robot localizes itself according to a colour sensor, where both motion and sensing are affected by a Gaussian error. The colour of the border is related to the number of particles in the cell.

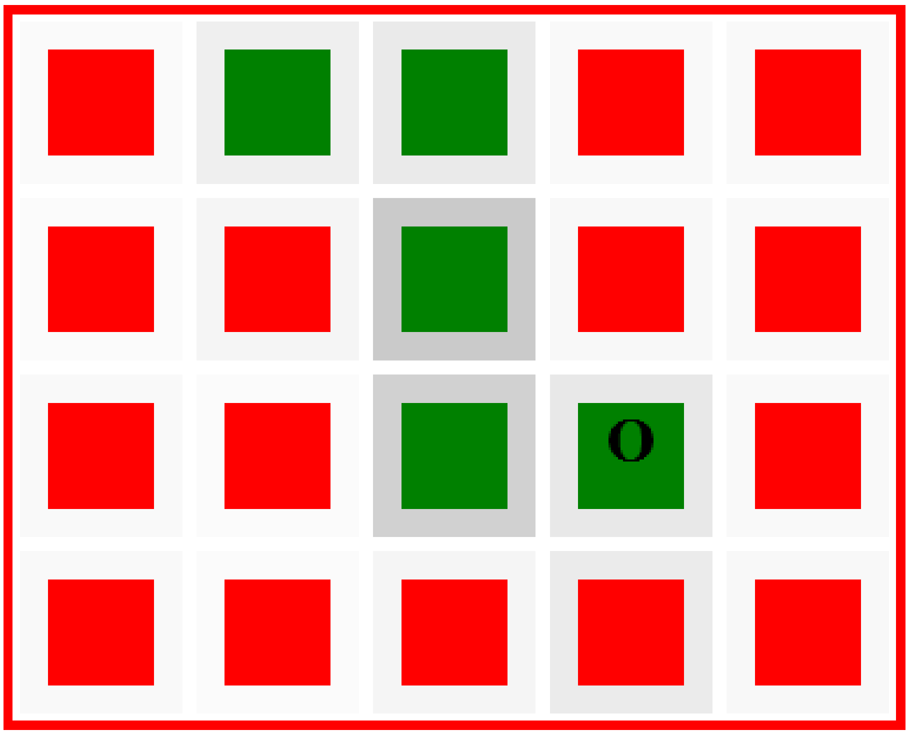


Figure 1 A sample 2D World, border darkness shows belief after some sense-update steps

2.3 The algorithm [4]

Initial distribution.

When the particle filter is initialized, whatever knowledge is available can be used to distribute the particles in the environment. In the most general case, particles are distributed at random over the entire environment.

```
N = <number_of_particles>
particles = []
```

```
for i in range(N):
  particles.append(createRandomParticle())
```

The update step.

Updating a particle means moving it according to the robot's motion:

```
m = <robot motion>
for i in range(N):
    particles[i].move(m)
```

and calculating its likelihood according to the current robot's measurements:

```
p = <robot percepts>
w = [] # likelihood of each particle
for i in range(N):
    w[i].append(particles[i].getLikelihood(p))
```

Resampling.

This is the crucial step of the particle filters, where a whole new set of particles is built:

```
new_particles = []
for i in range(N):
    sampled_part = pick_particle_with_prob (particles, w)
    new_particles.append(sampled_part.copy())
particles = new_particles
```

Determining the most likely location.

The belief of the robot about it's own location is given by the number of particles in each world cell:

```
world_cells = <portions of the environment>
for cell in world_cells:
   cell.probability = count_particles(particles, cell)
```

Taking errors into account.

The key feature that makes the particle filter work is uncertainty: the real robot has many uncertainties, due to motion, sensors' error, noise, external influences. The sensor's error is used in computing the `particle.getLikelihood(measurement)` function, and allows for giving some likelihood even to particles whose virtual measurements differ from the robot's own measurement. For instance, if the robot senses a landmark at 1 meter, it could be at 0.95, or 1.05, and particles in that range should be considered quite likely. But it could also be that the sensor is having a momentarily failure, so even particles at 2 meters from the landmark should be given some residual probability.

Error in the particle's motion update step not only reflect the uncertainty in the real robot's motion, but also help spread the particles over a local area: in the resampling step, many copies of a single original particle can be made, and they will all be at the orginal particle's location. But when they move with a Gaussian error, they will spread over a local area, creating new possible locations.

3 The case study: AIRobots

The demonstration is based on the simulator that was built for the AIRobots project [3]. This case study was chosen because the author is the development team of the simulator, and wanted to explore an alternative method of localizing the robot. The robot is an UAV, an autonomous flying robot built to inspect plants. When the robot is inside the plant, maybe inspecting a boiler or a chimney, it is out of sight and needs precise localization to accomplish its task. The map of the environment is known, and landmarks information can be perceived using artificial vision, distance sensors, lasers or other. The robot's location and orientation is actually made of six coordinates: x, y, z, yaw, pitch and roll, but pitch and roll can easily be read from the onboard sensors. Also, the robot stays horizontal, deviating from this pose only slightly and only for moving, therefore there are only four coordinates to be determined by localization.

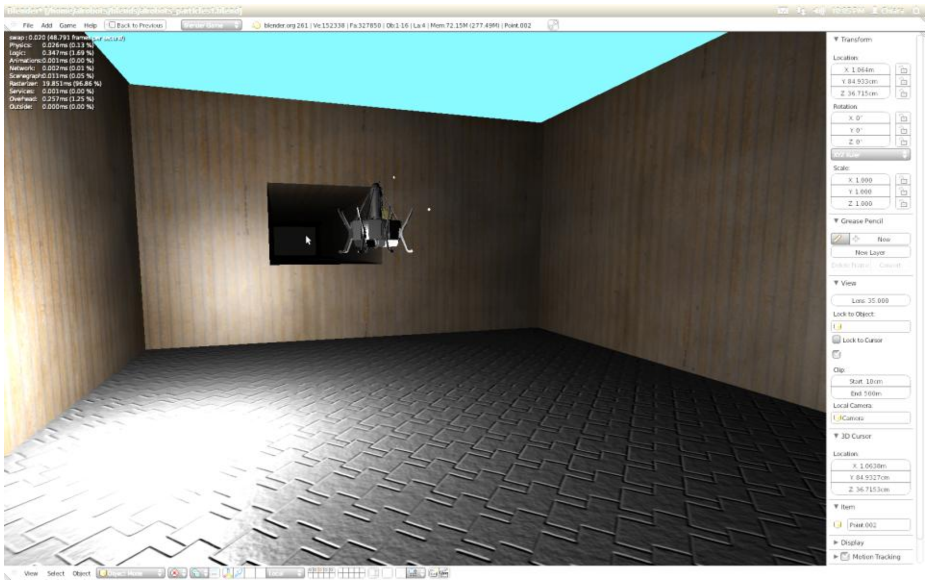
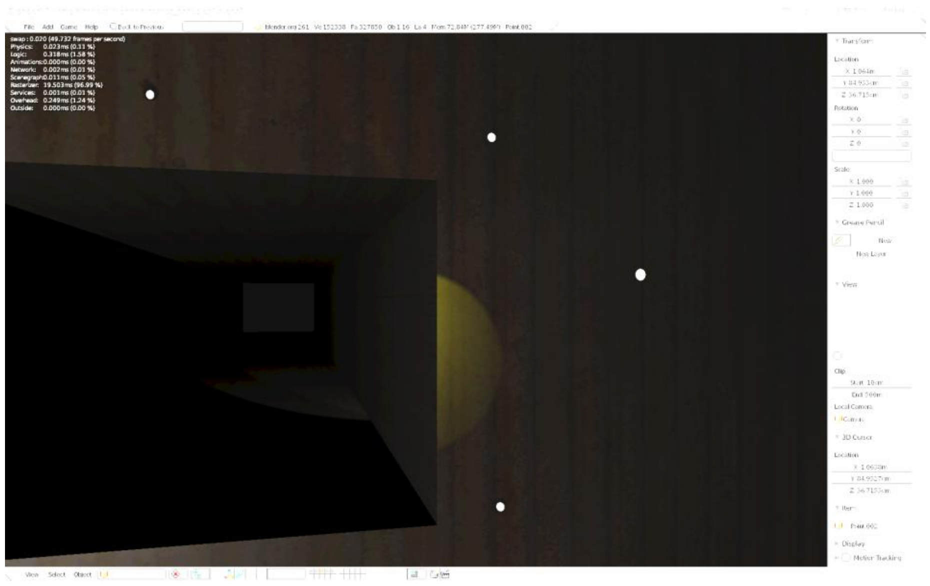


Figure 2 A dim simulation environment where the robot has little information

For this demonstration, the robot moves in a multiple-room environment, where the rooms have different sizes. The initial location of the robot is random, i.e. the robot doesn't have any clue to its relative position. Using a single low-res camera, the robot is able to individuate bright points, corresponding to lamps placed in the environment. Figure 3 shows a sample onboard view where bright dots are the lamps.



```
swap : 0.020 (49.732 frames per second)
Physics:     0.023ms (0.11 %)
Logic:       0.318ms (1.58 %)
Animations:0.000ms (0.00 %)
Network:     0.002ms (0.01 %)
Scenegraph0.011ms (0.05 %)
Rasterizer:  19.503ms (96.99 %)
Services:    0.001ms (0.01 %)
Overhead:    0.249ms (1.24 %)
Outside:     0.000ms (0.00 %)
```

Figure 3 Bright indicate the landmarks that can be seen from the onboard camera.

The demonstration shows how the robot moves in the "real" 3D environment and how its belief evolves as it moves, showing that from very simple information, like the points of the lamps in the camera image, the robot can infer all of the four coordinates.

4 Conclusions

Particle filters are a probabilistic method that can be used to localize a robot in a known environment, when very precise localization is necessary. The method is very easy to implement, with limited memory and computation demands, and works even when the information is scarce, like simple bright points on a camera image to infer four spatial coordinates.

5 References

1. S. Thrun; W. Burgard; D. Fox;, "Probabilistic robotic". MIT press 2006. Chapter 4.

2. S.J. Russell; P. Norvig;, "Artificial Intelligence - a modern approach. Third edition." Prentice Hall 2011. Chapters 13-15.
3. AIRobots project website: http://www.airobots.eu/. In particular Deliverable D4.2 for the simulator.
4. S. Thrun. CS 373 – Building a Robotic Car – Online course at: http://www.udacity.com/
5. F. Dellaert; F. Fox; W. Burgard; S. Thrun; "Monte Carlo localization for mobile robots". Proceedings from the IEEE international conference on Robotics and Automation. 1999
6. Turgut, B.; Martin, R.P.; , "Restarting Particle Filters: An Approach to Improve the Performance of Dynamic Indoor Localization," Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE , vol., no., pp.1-7, Nov. 30 2009-Dec. 4 2009
7. Sangjin Hong; Jinseok Lee; Athalye, A.; Djuric, P.M.; We-Duke Cho; , "Design Methodology for Domain Specific Parameterizable Particle Filter Realizations," Circuits and Systems I: Regular Papers, IEEE Transactions on , vol.54, no.9, pp.1987-2000, Sept. 2007