Simultaneous Localization and Mapping: A General Approach to Different Methods  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Progress Report III \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Veysel Erçağlar Yunus Atahan Uğur Can Kozan Melikcan Türkdemir

**I. INTRODUCTION**

Robots in millennium era were always popular. They were popular among both users and researchers. In mobile robots, self driving or observing from outside and processing inside were important. Under heavy research years, Simultaneous Localization and Mapping (SLAM) became extremely popular among researchers. SLAM is a method that on an unknown location, the agent is creating a map concurrently keeping the data of agent’s location. This technique allows a robot to behave like an intelligent being. SLAM is widely used in self-driving cars, and robots that built to make investigation on unknown places to people (Such as MARS). SLAM is preferred because with no prior knowledge robots are still making good progress. There are multiple SLAM algorithms on literature that are beneficial in particular case or not effective. Introduced algorithms for SLAM are as Kalman SLAM, EKF SLAM, FAST SLAM, L-SLAM, GraphSLAM, LSD-SLAM, S-PTAM, ORB-SLAM, MonoSLAM, CoSLAM. There are other algorithms used for SLAM but in this paper, we will try to focus on three of them. At the end of this paper, the implementations will show their comparisons in terms of their efficiency, run time complexity etc.

**II.METHOD**

**2.1 Particle Filter SLAM**

Particle Filter is a method that computing the posterior behavior in limited Markov Chains within discrete time. In a given time *t,* a state of Markov Chain is xt . Clearly, the state of xt is bounded to state xt-1 under regards of probabilistic law

Another state kt, that will be a stochastic projection of xt . Eventually, it will be generated by probabilistic approach: . In a generalist way of representation of estimation is and the measurement (update) is . Specific Kalman Filters are working under *O(d3*) run time complexity. “d” here is the given dimension space. Kalmans are for the cases where the Gaussian-Linear assumptions are appropriate for estimation. However, particle filters are in a generalist cases of partially unconstrained Markov Chains. The base structure is to estimate the posterior of a set of sample states } or particles. [1] denotes state of sample *i* and range varies between [1,*n*), *n* is the volume of particle filter. Particle Filters are working with the “Survival of the Fittest” concept. Each posterior is denoted with set of “weighted samples” Each particles is distributed randomly initially and their lifespan is decided by their weights. The generic pseudocode as follows:

**Fast SLAM**

Fast SLAM algorithm is introduced by Montemerlo et al. in 2002 as first successful implementation of Rao-Blackwellised particle filter that could handle large maps or real-world problems. Each landmark is represented by 2x2 EKF, therefore each particle must maintain M individual EKFs. In total, there are N·M EKFs, where M is the total number of particles in the particle filter and N is the total number of landmarks.

**Key Steps of Fast SLAM 1.0**

Fast SLAM algorithm draws samples according to standard odometry model being used to localization. It extends the path posterior by sampling a new pose for each sample.

In the next step, it computes the importance weight:

Q: measurement covariance

z: current observation

ẑ: expected observation (calculated for each individual)

As last step, it updates the belief of observed landmarks using the EKF update rule, then resamples using the standard resampling operation.

**Computational Complexity Fast SLAM Implementation**

Update robot particles: *O(N)*

Incorporate an observation into Kalman filters: *O(N log M)*

Resample particle set: *O(N log M)*

Total: *O(N log M)*

(*where N is the number of particles and M is the number of map features*)

**Fast SLAM 2.0**

Second iteration of Fast SLAM proposed by Montemerlo et al. in 2003, which considers the measurements during the sampling.

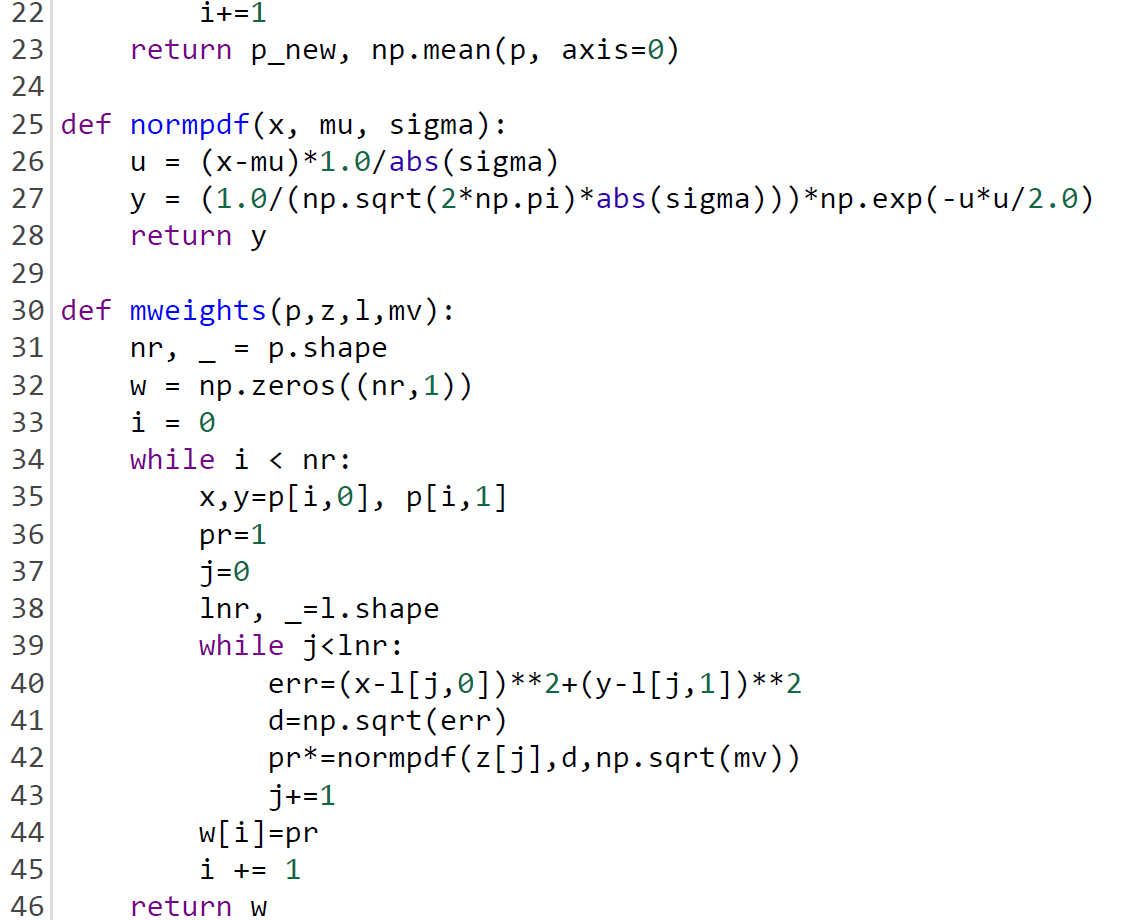
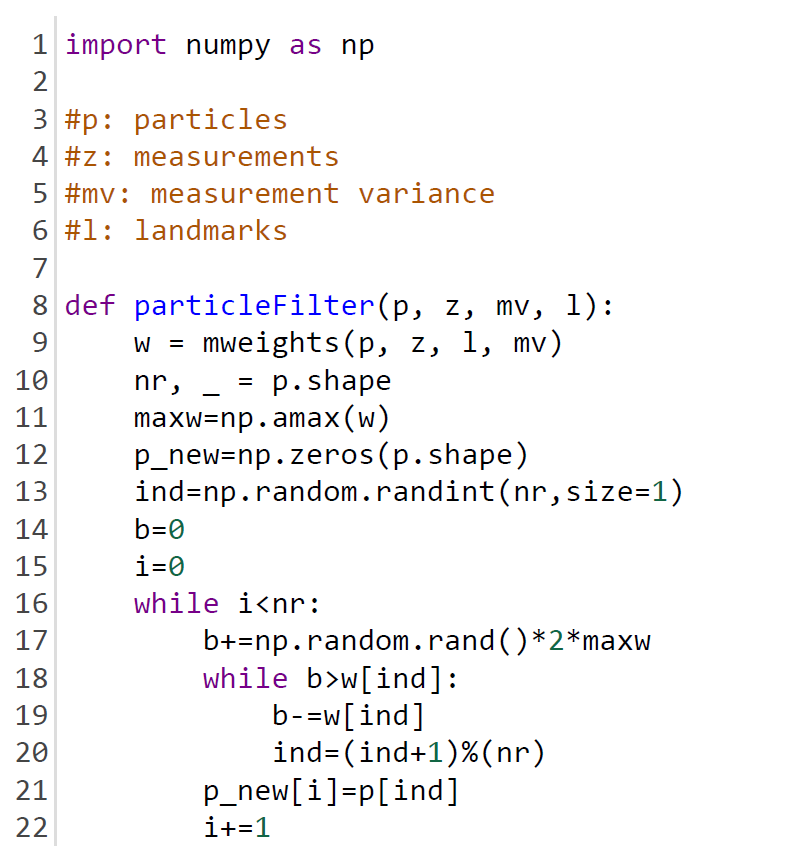
This leads to proposal distribution being more peaked around the true state where the system is in. As a result, less samples are needed. Compared to Fast SLAM 1.0, Fast SLAM 2.0 is more robust and accurate, however it is more complex.

-**Algorithm Particle Filter(*problem)*** *returns the resulting set of particles{  
 Initiate n many particles at time t=*t0 (Initial Time)  
 *Particle0 =Distribute initiated particles with respect to p(x0)* (Under Gaussian)  
 *While(t >0) {* Xt= *Create a particle for each previous state’s particle() from prediction  
 Distribute n particles from* Xt  *,each is distributed with probabilistic update )*

*}  
Return the outcome set of particles Xt  
}*

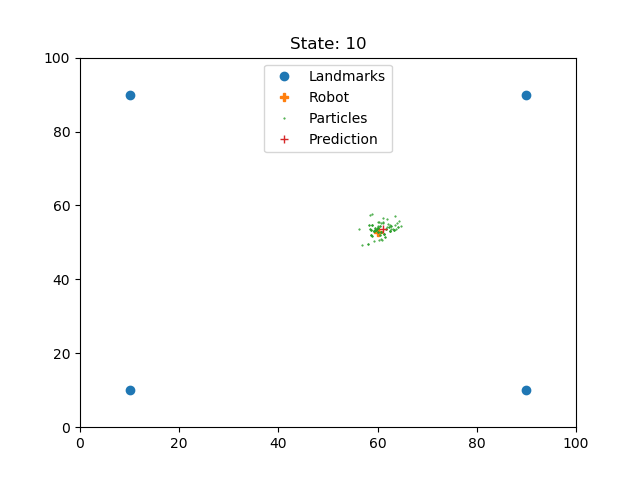
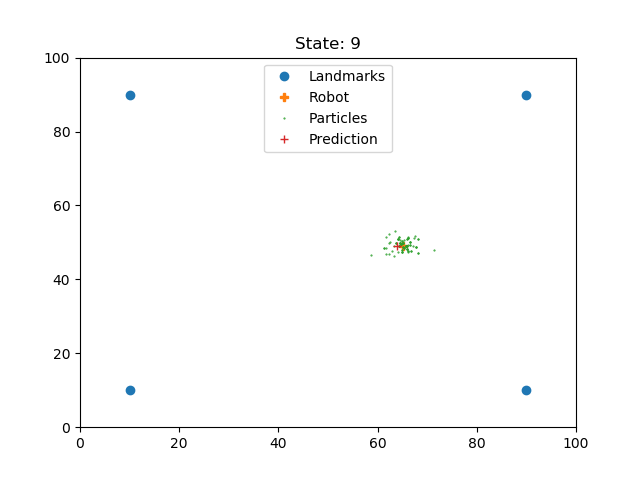
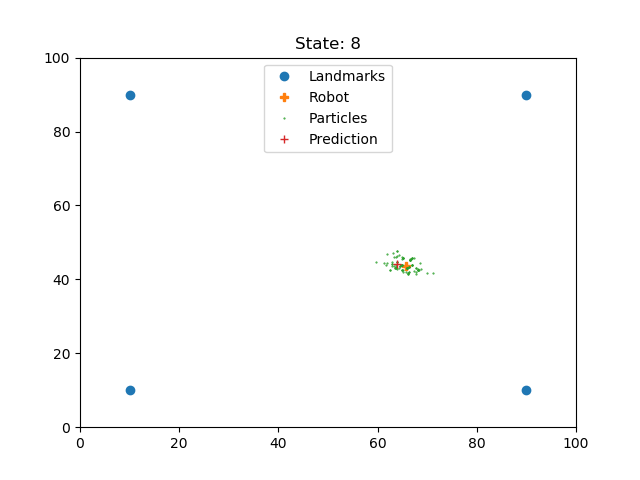
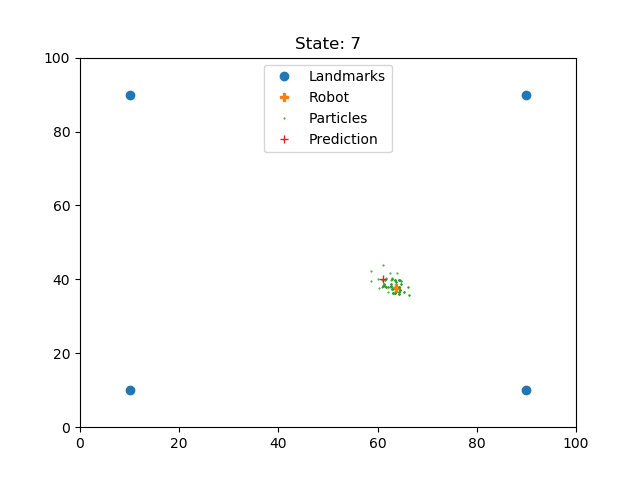
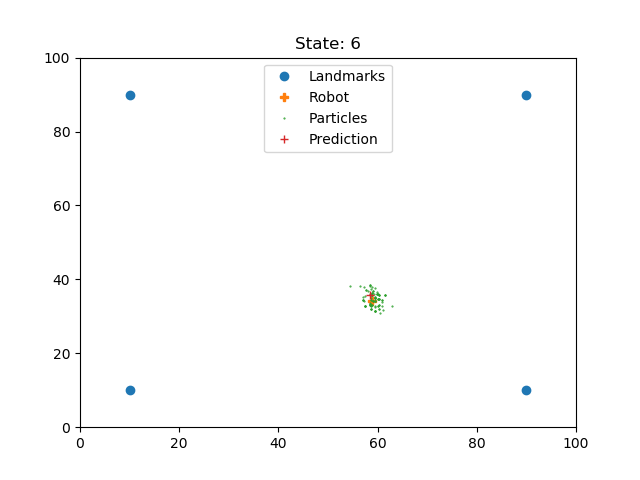
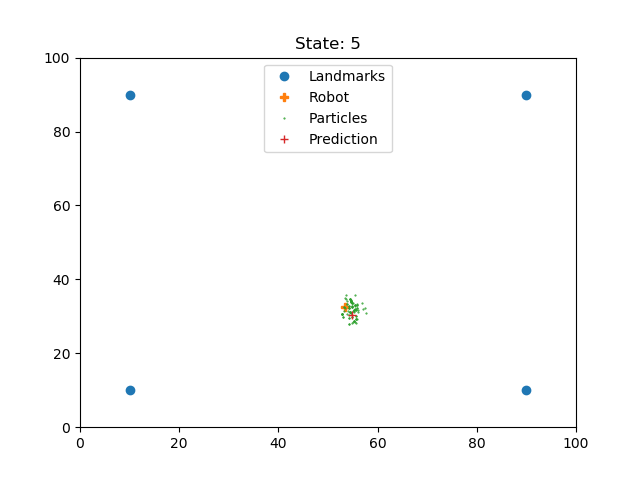
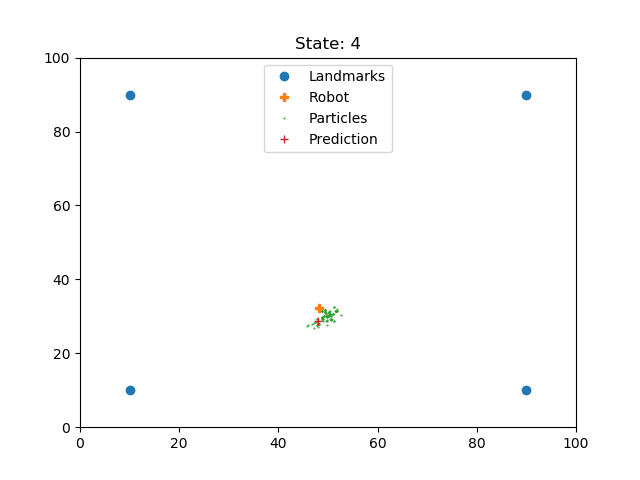
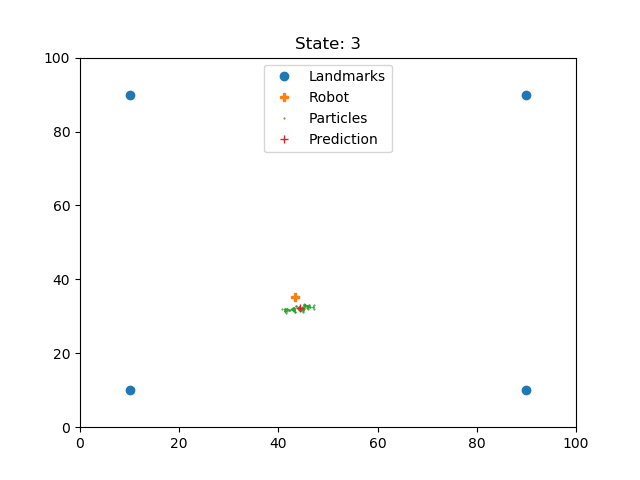
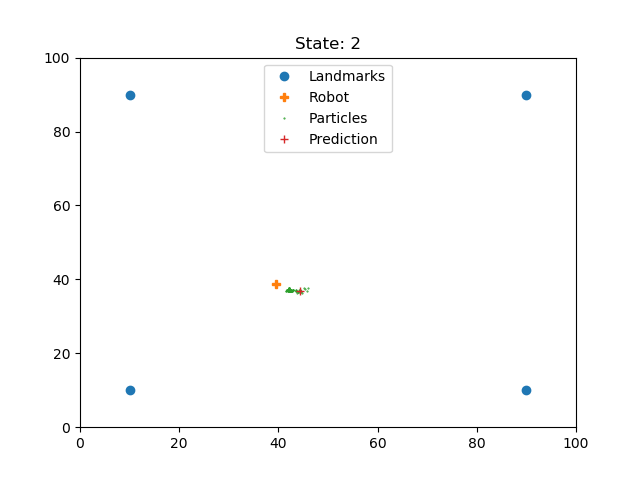
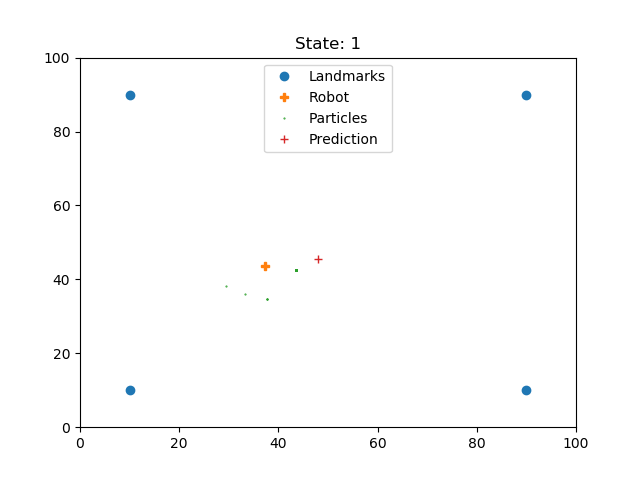
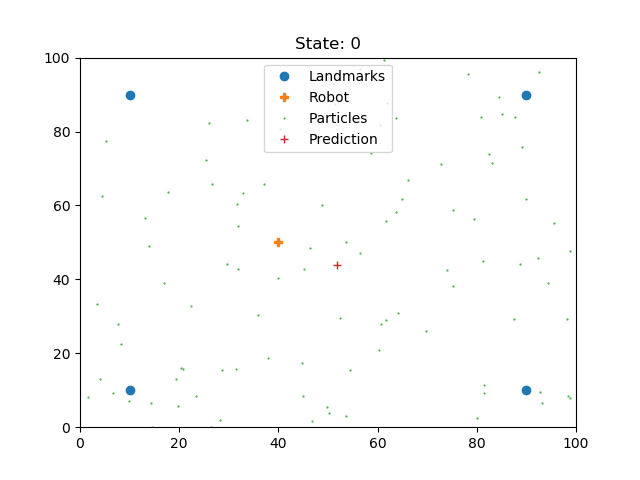
Eventually, a specialized particle filter algorithm will be used for SLAM. That is called Fast SLAM.

**Implementation of Particle Filter SLAM**



**III.Conclusion**

Results of above implementation with respect to states:

****

**REFERENCES**

[1] Thrun S. Particle Filters in Robotics, *In Proceedings of Uncertainty in AI (UAI)*

Carnegie Mellon University-Computer Science Department, Section 2, 2002.

[2] Çulha1 U., Turan B. *Particle Filter Based Fast Simultaneous Localization and Mapping*

Computer Engineering Department, Bilkent University,

[Cyrill Stachniss](Cyrill%20Stachniss).(2013.12.9*) SLAM Course - 12 - FastSLAM (2013/14; Cyrill Stachniss)* Retrieved from <https://www.youtube.com/watch?v=Tz3pg3d1TIo>