# Mobile Robot Localization

Patrick Wellins

**Abstract**—Obtaining the accurate location of a mobile robot is a non-trivial task due to uncertain sensor measurements and random variations in the environment. The Kalman Filter and Adaptive Monte Carlo Localization (in short AMCL) are used to localize mobile robots in real time. Both the Kalman Filter and AMCL are probabilistic algorithms that can map noisy sensor and odometry data into accurate estimates of a robots position. This report shows the results of localizing two mobile robots in a simulated environment using AMCL. The localization and navigation results of the two robot models are compared to generate experimental data that can provide insights into building improved systems for autonomous robots. Further discussion is provided on the topics of localization algorithm selection and commercial applications.

**Index Terms**—Robot, IEEEtran, Udacity, LaTeX, Localization.

## 1 Introduction

Localization is a key aspect of developing autonomous robots and vehicles. For example, a mobile robot in a warehouse may be given the task to pick up and drop off a palette at specific locations represented by x-y coordinates. Monte Carlo localization (MCL) and Kalman filters are two algorithms that can be used to solve this task. A notable and early example of localization is the use of the Kalman filter in the Apollo navigation system [1]. Localization can be broken into different classes which include position tracking, global localization, and the kidnapped robot problem. Position tracking is when a robot is localized when the initial pose of the robot is known. Global localization is the problem of localizing when the initial position is unknown. The kidnapped robot problem is regaining the location of a robot when the robot is randomly moved from one location to another in the environment.

This report examines the results of localizing two mobile robot models in simulation with AMCL. The mobile robots were equipped with lasers to measure their location in the environment, and each robot followed a non-linear path to the goal point. Various parameters were tuned to achieve successful navigation and localization.

## 2 Background

The objective that is defined for the experiment is successful localization and navigation of two mobile robots in simulation. AMCL was employed for localizing the two mobile robots. Intelligent selection of a localization algorithm is important given the different profiles of MCL and the EKF. Background information on EKF's and MCL is provided as a guide to algorithm selection.

### 2.1 Kalman Filters

The Kalman Filter comes in different varieties, the two which will be discussed are the linear Kalman filter and the Extended Kalman filter (EKF in short). The Kalman filter models the state of a robot with a Gaussian distribution. In the case where more than one variable is predicted, a multivariate Gaussian is used for state estimation. It is important to note that Kalman filters are used for position tracking and therefore are not used to solve global localization.

To illustrate position tracking, imagine a car that is traveling linearly. An initial estimate of the position is represented by a Gaussian distribution. An uncertain measurement of position is taken from the car's sensors generating another Gaussian distribution. The parameters (mu and sigma) of the prior state and the current measurement are used to form updated parameters for the Gaussian posterior distribution which estimates position. This posterior estimates position with more certainty and accuracy than the prior probability distribution of the location. A limitation of the linear Kalman filter is that is can only handle states that change linearly. The EKF can track the position of non-linear state transitions; and does so by creating a linear approximation function with the first two terms of the function's Taylor series. The linear approximation of the state transitions and measurement functions preserves the Gaussian distribution estimate of position.

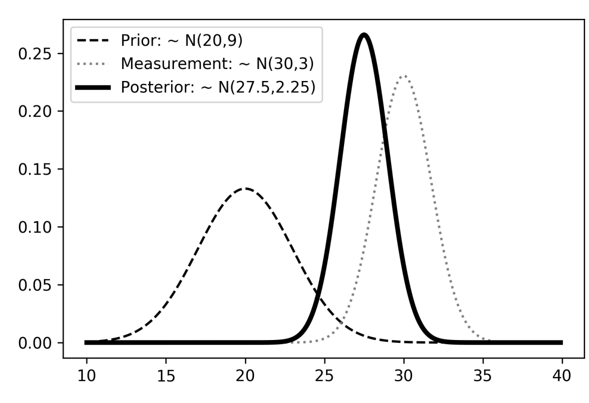


Fig. 1. The estimation of car location from initial estimate, measurement, to posterior.

#### 2.1.1 1-D Kalman Filter Parameter Updates

$\mu' : PosteriorMean \;\; \sigma^{2'} : PosteriorVariance$
$\mu : PriorMean \;\; \sigma^2 : PriorVariance$
$v : MeasurementMean \;\; r^2 : MeasurementVariance$

$$\mu' = \frac{r^2\mu + \sigma^2 v}{\sigma^2 + r^2}$$

$$\sigma^{2'} = \frac{1}{\frac{1}{\sigma^2} + \frac{1}{r^2}}$$

### 2.2 MCL and AMCL

The MCL algorithm initially generates particles that are uniformly spread throughout the mapped environment. The uniform distribution represents a distribution that reflects no knowledge of the robot's state. Once the robot makes a sensor measurement, each particle is assigned a weight that is proportional to the probability of the measurement given the state of the particle. The particles are then re-sampled with replacement to generate a new set of particles that better estimate the robots position. The particles then move in the direction of the robot's motion with some embedded noise in the particle movement. This process repeats and the particles are continually re-sampled to update the posterior distribution of the robots pose. If MCL is successful the particles converge to the actual state of the robot.

### 2.3 Comparison and Contrast of Algorithms

The following two sections describe the strengths and weaknesses of each algorithm, as well as the circumstances that each algorithm is best suited for.

#### 2.3.1 Characteristics of EKF's

The EKF is advantageous to use when the problem to be solved is position tracking. If the initial state of the robot is known and the sensor noise is Gaussian, the EKF is a good choice due to the efficiency of the algorithm with respect to time and space. The time required for each update is $O(K^{2.4}+n^2)$ where $K$ is the dimension of the measurement vector and $n$ is the dimension of the state vector $x_t$ [2]. If the computational resources of a robot are limited and the previously stated constraints are satisfied, the EFK is a good choice for localization.

#### 2.3.2 Characteristics of Particle Filters

MCL and AMCL are more flexible than Kalman filters in a number of ways. MCL and AMCL can create posteriors that match any distribution, and the distribution of measurement noise is not restricted. AMCL can solve global localization and the kidnapped robot problem making it robust and practical. If global localization is a requirement, then AMCL is an easy choice over the EKF. However, there is a trade off between accuracy and computational cost when employing AMCL; increasing accuracy through additional particles comes with increased computational cost.

## 3 Simulations

In simulation, two mobile robot models navigated to a goal position and both were localized with AMCL. The fist mobile robot is referred to as the benchmark model; the benchmark model is a simple two wheeled robot design with a camera and hokuyo laser sensor. The second robot which is referred to as the personal model is larger and has a more complex design than the benchmark model. The goal for both models is to be accurately localized and reach the goal destination.

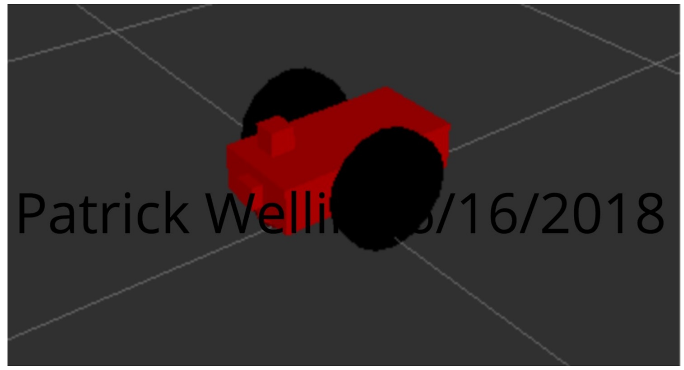


Fig. 2. Benchmark model.

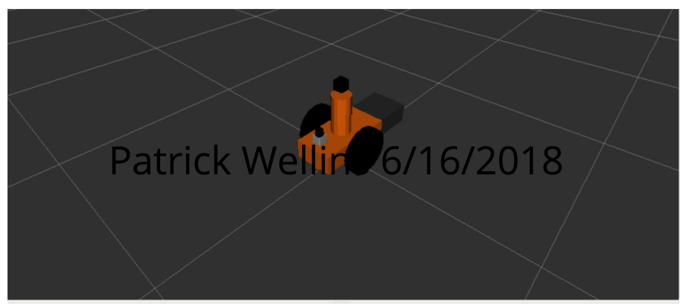


Fig. 3. Personal Model

### 3.1 Achievements

Both the benchmark model and personal model were able to navigate to the goal position; AMCL also did an excellent job of localizing both models during the route and at the end goal. Figures 4, 5, and 6 show the green particles tightly bunched under both models.

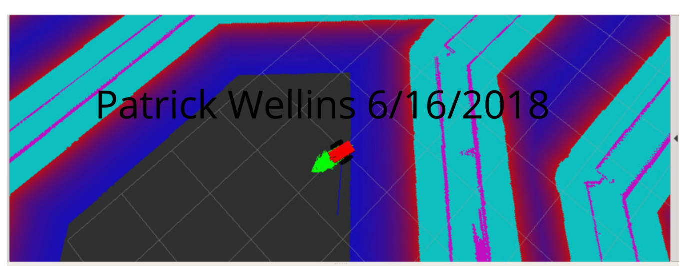


Fig. 4. The benchmark model at the goal position with the global cost map displayed.

## 3.2 Benchmark Model

### 3.2.1 Model design

Table 1 and Table 2 contain information on the links and joints of the two wheeled benchmark robot.

TABLE 1
Benchmark Robot Links

| Links | Geometry |
|---|---|
| Hokuyo | box size="0.05 0.05 0.05" |
| Camera | box size=".05 .05 .05" |
| Chassis | box size=".4 .2 .1" |
| Left Wheel | cylinder length="0.05" radius="0.12" |
| Right Wheel | cylinder length="0.05" radius="0.12" |

TABLE 2
Benchmark Robot Joints

| Joints | Origin |
|---|---|
| Hokuyo Joint | origin xyz="0.15 0 0.1" rpy="0 0 0" |
| Camera Joint | origin xyz="0.2 0 0" rpy="0 0 0" |
| Left Wheel Hinge | origin xyz="0 0.15 0" rpy="0 0 0" |
| Right Wheel Hinge | origin xyz="0 -0.15 0" rpy="0 0 0 |

### 3.2.2 AMCL Package

The AMCL package allows for the mobile robots to be localized in the environment. The AMCL node subscribes to various topics including laser scans, tf, initial pose, and the map. The laser scans allow for the detection of objects in the environment. The tf topic allows for tracking of various coodinate frames over time. The initial pose is what puts the moble robot in its initial position, and the map topic communicates information about the environment map. The AMCL also publishes to the amcl pose, particle cloud, and tf topics. The particle cloud topic contains the set of pose estimates that are generated from the amcl algorithm.

TABLE 3
AMCL Node

| Topic | Publish/Subscribed |
|---|---|
| scans(Laser scans) | Subscribed |
| tf Transforms | Subscribed |
| initialpose | Subscribed |
| map | Subscribed |
| amcl pose | Published |
| particle cloud | Published |
| tf | Published |

TABLE 4
move base Node

| Topic | Publish/Subscribed |
|---|---|
| move base goal | Subscribed |
| move base cancel | Subscribed |
| move base feedback | Published |
| move base status | Published |
| move base result | Published |
| cmd/vel | Published |

### 3.2.3 Move Base Package

The move base package is what allows the mobile robot to take actions in the environment to reach a goal. Move base is subscribed to the goal and cancel topics. The goal topic provides a goal for the mobile robot to move to. The cancel topic contains messages that allow for cancellation of a goal. The move base package publishes to the feedback, status, result, and cmd/vel topics. The cmd/vel topic contains a stream of velocity commands that are to be executed by the mobile robot.

### 3.2.4 AMCL Parameters

The minimum and maximum particle values are 100 and 1000 respectively, the number of particles creates the distribution that estimates the pose of the mobile robot. The (update min d) and (update min a) parameters control the translational and rotational movement that is required before a filter update is performed. The laser min and max range controls the range that the laser can sense in the environment. The odom alpha 1-4 parameters control the amount of estimated noise from odometry translation and rotation. The transform tolerance parameter controls the latency of the tf information.

### 3.2.5 Move Base and Cost Map Parameters

Many of the parameter values for the move base and cost map come from the paper ROS Navigation Tuning Guide by Kaiyu Zheng [3]. The obstacle range parameter for the benchmark model is set to 0.9; if an obstacle is within 0.9 meters of the robot it will be added to the costmap. The raytrace range is set to 2.5, which means free space will be within a 2.5 meter range. The inflation radius is set to 1.5, 1.5 meters is the minimum distance between the robot and the obstacles. The cost scaling factor of 2.58 creates a costmap with costs that rise gradually as the robot approaches obstacles, this makes the robot prefer paths that are far from obstacles. The path distance bias determines how much the robot should follow the global path; this parameter is set to 32, which is the default setting. The goal distance bias determines the weight for how much the robot should track its local goal; this parameter is set to 20 which is less than the default setting of 24. The occ dist scale parameter controls the weight for how much the robot should attempt to avoid obstacles; this parameter was set to 0.02 which is higher than the default setting of 0.01, this indicates the robot is less flexible to go near obstacles.

TABLE 5
AMCL Parameters

| Parameter | Value |
|---|---|
| min particles | 100 |
| max particles | 1000 |
| update min d | 0.01 |
| update min a | 0.01 |
| laser max range | 5 |
| laser min range | 0.2 |
| odom alpha 1-4 | 0.1 |
| transform tolerance | 0.2 |

TABLE 6
Move base Parameters

| Parameter | Value |
|---|---|
| obstacle range | 0.9 |
| raytrace range | 2.5 |
| inflation radius | 1.5 |
| cost scaling factor | 2.58 |
| path distance bias | 32 |
| goal distance bias | 20 |
| occ dist scale | 0.02 |
| transform tolerance | 0.2 |
| sim time | 4.0 |
| cost factor | 0.55 |
| neutral cost | 66 |
| lethal cost | 255 |

### 3.3 Personal Model

#### 3.3.1 *Model design*

The personal model is a modified version of the benchmark model. The camera sits on top on a cylinder that is has a height of 0.3 meters and a radius of 0.05 meters. This platform extends the observable horizon of the camera view. The gray square cargo area at the back of the robot is for carrying loads. This cargo area is a three dimensional rectangle has a length and with of 0.2 meters and a height of 0.1 meters. A caster sits beneath the cargo platform for stabilization. The cargo platform is connected to the chassis with a fixed joint.

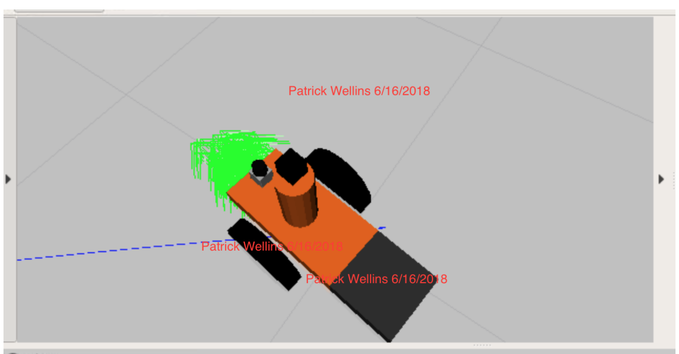


Fig. 5. A close up view of the custom mobile robot model. The camera sits on a cylinder and a gray cargo platform is attached to the back of the robot.

#### 3.3.2 *Packages and Parameters Used*

The Move Base and AMCL packages were used to navigate and localize the personal model. The parameters for these packages were not modified to successfully localize and navigate the custom model in the environment. The parameters for the personal are displayed in tables 5 and 6.

## 4 Benchmark Model Results

The benchmark model was localized after 5 seconds of initial movement, and reached the goal location in 1 minute 32 seconds. There was an initial loop that the benchmark model took before it started to follow the global path to the goal position. The benchmark model moved in a smooth path most of the route, with the exception being the 180 degree turn that is necessary to take to exit the corridor. Localization results are very accurate given that the particles are very tightly clustered under the benchmark model's base as shown in Fig 2.

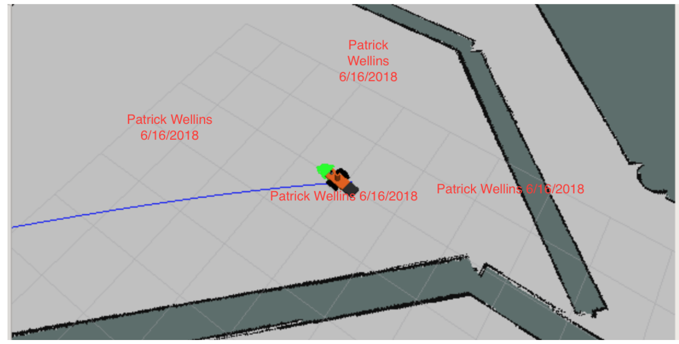


Fig. 6. The custom robot is at its goal position. The particles are tightly converged on the goal point. AMCL was effective for the custom model without modifying any of the parameter values that were used for the standard mobile robot model.

## 5 Personal Model Results

The person model was localized after 6 seconds of initial movement, and reached the goal position in 1 minute 26 seconds. There was an initial loop that the personal model took before it started to follow the path to the goal position. The personal model moved in a smooth path most of the route, with the exception being the 180 degree turn that is necessary to take to exit the corridor. Localization results are very accurate given that the particles are very tightly clustered under the custom model's base.

### 5.1 Technical Comparison

The personal model is a larger mobile robot than the benchmark model and the personal model's camera is placed on a platform. The additional size of the personal model did not necessitate any changes in parameter values that were used for the benchmark model. This indicates that the parameter values are likely robust for larger robots. For the benchmark model it took 5 seconds for the particles to converge, and 1 minute 31 seconds to reach the goal location. For the custom model it took took 6 seconds for the particles to converge and 1 minute 26 seconds to reach the goal. Since the results with regards to localization and navigation are so close, it is difficult to say with confidence which model had better performance. The benchmark model localized in 5 seconds versus 6 for the personal model, this could imply that smaller models are easier to localize. It is plausible that smaller models would navigate better due to increased capability to stay in low cost regions on the map.

## 6 Discussion

The benchmark and personal models had excellent results for localization. The particles converged to very tight regions under the bases at the goal positions. The particles also converged quickly to both robots positions after initial movement, taking less than 6 seconds to converge for both models. Though both robot models reached the target in a respectable period of time, both models struggled with navigation at certain points. During the majority of the route navigation was respectable, however when the robots initially move they go in the opposite direction that they should move. This is likely due to the initial position being in a tight corridor, this likely causes the robots to move to a space that has a lower cost value on the map. Further tuning the move base parameters is likely necessary to achieve better navigation results. The path distance bias controls how close the local planner stays to the global path [3], increasing this can potentially solve the problem of the initial loop that does not follow the global path. A smaller model could potentially be a solution to the initial wasted movement; the base would be small enough where it would be within a region of low cost that would not facilitate movement to a lower cost region that is further from the goal. Decreasing the values (of update min d) and (update min a) made a big improvement in compressing the region of the particles. These variables control the frequency of filter updates that are dependent on translational and rotational movement.

### 6.1 Kidnapped Robot Problem

AMCL can solve the kidnapped robot problem by randomly injecting particles into the particle sets [2]. This can handle the problem of particle deprivation, a condition when there are no particles close to the actual location of the robot. If the robot is picked up and moved the current particle cluster will be predict an inaccurate location, however randomly generated particles can be located near the true location and these particles will have a higher likelihood of being re sampled and regenerating a particle set that accurately localizes the robot.

### 6.2 Value of Localization and Industry Applications

Localizing robots is a valuable tool in an industrial setting for a number of reasons. A localized mobile robot in a warehouse can automate moving heavy and bulky objects from location to location. Mobile robots can be used to navigate in environments that are very dangerous to humans such a nuclear reactors and under ground mines. If these robots can be localized, they can go to specific locations to transport materials such as coal or handle tasks in areas with high levels of radiation.

## 7 Conclusion / Future work

The objective of localizing two mobile robot models in simulation with AMCL was achieved. The two models also navigated to the goal postion successfully. Further parameter tuning for the move base package can improve navigation further. Due to time constraints these improvements were not thoroughly pursued, the subject of tuning move base parameters to optimize localization is a worthy topic to delve in more deeply. If these parameters can be set in a systematic way it would greatly reduce the time necessary to navigate mobile robots, having a theoretical basis for setting parameters with experimental evidence would be superior to the trial and error technique.

### 7.1 Improving Base Dimensions and Sensor Layout

Creating a base that is more aerodynamic is something that can be pursued to create better mobile robots. Improved aerodynamics can improve handling, and can also result in a more energy efficient robot. There are so many tasks that a mobile robot can carry out and each task will require different designs. Sensors should be placed to maximize perception of the environment. Sensors should not be placed in areas that limit perception or areas that are likely to be damaged when the mobile robot is moving. If sensors can be placed on links with joints, the sensors can move to orient themselves in positions that are best suited for the environment.

### 7.2 Hardware Deployment

More particles results in more accurate localization at the cost of computational complexity [2]. The time complexity of AMCL is linear in proportion to the number of particles. Faster hardware will improve the performance of AMCL, as seen in the simulations, re sampling more frequently resulted better localization. The NVIDIA Jetson TX2 would be a good choice for high performance, since it provides server class AI compute performance .

## References

[1] M. S. Grewal and A. P. Andrews, "Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]," *IEEE Control Systems*, vol. 30, pp. 69–78, June 2010.
[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.
[3] K. Zheng, "ROS navigation tuning guide," *CoRR*, vol. abs/1706.09068, 2017.