Fast and Robust Localization using Laser Rangefinder and WiFi Data

Renato Miyagusuku, Yiploon Seow, Atsushi Yamashita and Hajime Asama

***Abstract*— Laser rangefinders are very popular sensors in robot localization due to their accuracy. Typically, localization algorithms based on these sensors compare range measurements with previously obtained maps of the environment. As many indoor environments are highly symmetrical (e.g., most rooms have the same layout and most corridors are very similar) these systems may fail to recognize one location from another, leading to slow convergence and even severe localization problems. To address these two issues we propose a novel system which incorporates WiFi-based localization into a typical Monte Carlo localization algorithm that primarily uses laser rangefinders. Our system is mainly composed of two modules other than the Monte Carlo localization algorithm. The first uses WiFi data in conjunction with the occupancy grid map of the environment to solve convergence of global localization fast and reliably. The second detects possible localization failures using a metric based on WiFi models. To test the feasibility of our system, we performed experiments in an office environment. Results show that our system allows fast convergence and can detect localization failures with minimum additional computation. We have also made all our datasets and software readily available online for the community.**

I. Introduction and Related Work

Monte Carlo Localization (MCL) is among the most popular localization algorithms used in robotics [1]. Having as its most appealing characteristics its ease of implementation and its good performance across a broad range of localization problems. MCL can be used with a plethora of sensors by changing its perceptual model to fit the desired sensor. Previous works have use sonars [2], laser rangefinders [3], color cameras [4], WiFi [5], among others.

In this work we propose a framework, W-LRF, which incorporates the advantages of WiFi-based localization into a typical MCL that uses laser rangefinders. Figure 1 shows the main components of our approach. We incorporate WiFi data in order to achieve faster convergence and higher robustness. In particular, we are interested in: (i) how to combine range and WiFi data in order to achieve fast and reliable global localization; and (ii) how to detect localization failures once the system has converged. We achieve (i) by initializing our MCL using samples taken directly from the WiFi posteriors with constraints given by the environment's occupancy grid map; and (ii) by monitoring WiFi particle weights.

Laser rangefinders are quite popular due to the accuracy of their solutions. However, they have two main issues: they are susceptible to the data association problem and MCL systems using only laser rangefinders (LRF-MCL) have slow convergence. The data association problem refers to the misclassification between confusable features (e.g., failing to distinguish between two rooms with similar scans or different corners in a room), and can lead to severe localization problems. LRF-MCL convergence is slow due to symmetries in the environment, e.g., measurements at several spots in a long corridor may be identical, or several rooms in an office building may have the same outlay.

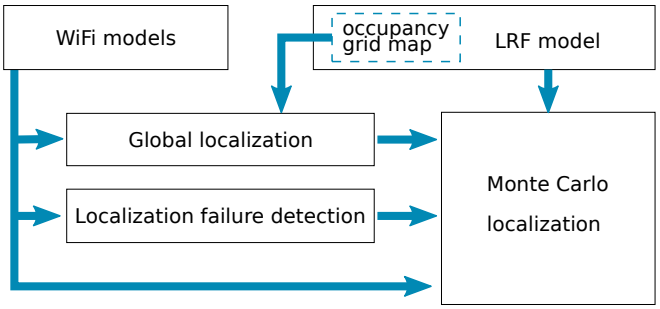


Fig. 1. Overview of our proposed approach and its two main components (i) global localization module and (ii) localization failure detection.

Using WiFi data is a less popular method for localization. These methods employ WiFi signal strength previously acquired in an environment to predict robot's location. Either by matching new measurements to the most similar samples in their training dataset [6]–[8]; or by learning location-signal strength mappings, and computing the likelihood of locations given new measurements and these mappings [9]–[11]. Although WiFi-based localization systems do not achieve as high accuracy as those based on laser rangefinders, they are immune to the data association problem and converge quickly. WiFi data has no data association problem, as all IEEE802.11 compliant WiFi packets transmit their sources' own unique identifier (macaddress) as part of its message header, identifying themselves unequivocally. And, although several locations in an environment may have the same signal strength values for a given access point, due to the sheer number of access points in modern buildings (ranging from several tens to few hundreds) symmetries are unlikely, allowing fast convergence. Although these characteristics make their usage appealing in a multi-sensor setting, few researchers have used WiFi data in such manner, as most research has concentrated in WiFi only approaches. Previous works in the area include the work by Biswas *et al.* [12], where a multi-sensor system composed by a laser rangefinder, a depth camera and WiFi data was proposed. In this approach, each sensor was processed independently

This work was funded by Tough Robotics Challenge, ImPACT Program of the Council for Science, Technology and Innovation (Cabinet Office, Government of Japan).

All authors are with the department of Precision Engineering, Graduate School of Engineering, The University of Tokyo, Japan. {miyagusuku, seow, yamashita, asama}@robot.t.u-tokyo.ac.jp

using different localization algorithms; with the robot choosing, at each location, the localization algorithm's output with the least variance as the system's output. The work by Ito *et al.* [13], where a system composed by a color and depth camera, and WiFi data was proposed; with WiFi data being used only for initialization of the localization algorithm that used the color and depth camera. As well as our previous work [14] where we presented a system composed by a laser rangefinder and WiFi data, where WiFi data was used to detect localization failures by comparing a set of particles sampled for each new WiFi measurement and MCL's particles. Contrary to our previous work, our new metric does not require constantly sampling the sensor posteriors, which is computationally expensive.

II. MCL AND SENSOR MODELS

MCL is considered as a global localization technique, as it can solve pose estimation without knowing the initial pose, by addressing the problem under global uncertainty. The more robust implementations also solve the so-called *kidnapped robot problem*, in which a robot is carried to an arbitrary position during its operation without its knowledge. This problem is considered much more difficult than global localization, as the robot might firmly believe to be somewhere else at the time of the kidnapping. This problem is commonly used to test a robot's ability to recover from catastrophic localization failures and it can arise in practical robot applications from poor odometry, collisions, slippage, and/or wrong sensors' measurements. In our work, we design a system that is capable of fast global localization, and is resilient to the kidnapped robot problem.

A. MCL

An MCL is essentially a particle filter, which is an implementation of the Bayes filter, combined with probabilistic models of robot perception and motion. MCL recursively estimates the posterior distribution of the robot's pose $\mathbf{l}$ given sensor observations $\mathbf{s}$ as,

$$p(\mathbf{l}_t) \propto p(\mathbf{s}_t|\mathbf{l}_t) \int p(\mathbf{l}_t|\mathbf{l}_{t-1}, u_t) p(\mathbf{l}_{t-1}) d\mathbf{l}_{t-1}, \qquad (1)$$

which is the basic equation for all Bayesian filters. With $p(\mathbf{s}_t|\mathbf{l}_t)$ being the probability of the observation $\mathbf{s}_t$ at position $\mathbf{l}_t$ computed by the robot perception model; and $p(\mathbf{l}_t|\mathbf{l}_{t-1}, u_t)$ the next state transition probability as computed by the robot motion model, with u_t being velocity commands or most commonly odometry information.

For this work, given that planar motion is considered, the robot's pose $\mathbf{l}_t$ consists of the robot's position in a x-y Cartesian coordinate system and its heading direction θ - i.e., $\mathbf{l} = [x\ y\ \theta]$. MCL represents $p(\mathbf{l}_t)$ by a set of s weighted particles $p(\mathbf{l}) = \{\mathbf{l}^{(i)}, w^{(i)}\}^{i=\{1,\dots,s\}}$; where $\mathbf{l}^{(i)}$ is a pose and $w^{(i)}$ is a non-negative scalar. These weights are calculated using the robot's perception model, i.e., $w^{(i)} = p(\mathbf{s}_t|\mathbf{l}_t^{(i)})$. After incorporating odometry information to compute the next state transition probability, MCL resamples this set of particles according to these weights. So the density of the particles within an area is proportional to the probability of the robot being in the vicinity of that area. Making the set of particles to converge to the robot's true pose with time.

An important parameter in MCL is the number of particles s, which can be either fixed or variable. For our implementation we set the initial, maximum, number of particles and use KLD sampling [15] to reduce the number of particles over time as MCL converges, requiring less particles to accurately describe the robot's pose posterior.

As two different sensors are employed, our system computes two different perceptual likelihoods: $p(\mathbf{r}|l)$ for range data measured using the system's laser rangefinder, and $p(\mathbf{z}|l)$ for received signal strength (RSS) data measured using the systems WiFi network interface controller.

B. LRF perceptual model

For the LRF perceptual model, we employ the *likelihood field model* as described in [1]. This model computes the probability of each range measurement to hit an obstacle given the robots pose and a map of the environment. Other than the probability of hitting an obstacle due to the robots location in the map, the model considers false measurements due to sensor noise and random failures. Combining these three distributions the model finally yields the sensor model $p(\mathbf{r}|\mathbf{l}, map)$.

C. WiFi perceptual model

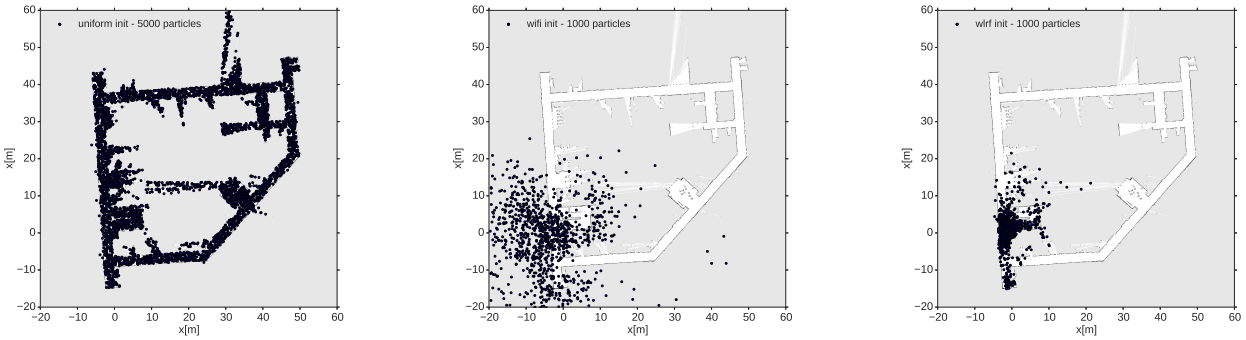
For the WiFi perceptual model, in this work we learn location-signal strength mappings using Gaussian Processes and path loss models, as described in [5]. The most important assumption of this approach is that heading direction does not affect signal strength information. While this is not strictly true, as most antennas have non-uniform radiation patterns, it holds well in practice.

Our approach assumes a training dataset $(\mathbf{X}, \mathbf{Z})$, of n data pairs $(\mathbf{x}_i, \mathbf{z}_i)$ has been previously collected. Where $\mathbf{x}_i \in \mathbb{R}^2$ are x-y Cartesian coordinates and $\mathbf{z}_i \in \mathbb{R}^m$ with $\mathbf{z}_i = [z_{(i,0)}, \cdots, z_{(i,m-1)}]$ are the RSS measurements from m different access points collected at $\mathbf{x}_i$. Under the GPs approach, it is assumed that the correlation of any two output values can be described as a function of their input values. Given this assumption, for any finite number of data points, the GP can be considered to have a multivariate Gaussian distribution, and therefore be fully defined by a mean function $m(\mathbf{x})$ and a kernel function $k(\mathbf{x}_p, \mathbf{x}_q)$.

For our approach we chose path loss functions (which is a parametric function which is a simplification of the physical phenomena of electromagnetic wave propagation through space) as the mean functions. Specifically we chose functions of the form,

$$PL(\mathbf{x}) = k0 - k1 \log(d) + \epsilon_{pl}, \qquad (2)$$

with d being the Euclidean distance between the position where the RSS measurements were taken ($\mathbf{x}$) and the predicted position of the access points $(apx,\ apy)$, $k0$ and $k1$ positive constant and ϵ_{pl} a Gaussian noise with variance σ_{pl}^2. Path loss parameters ($\theta_{pl} = [k0,\ k1,\ apx,\ apy, \sigma_{pl}^2]$) are



(a) Initialization from an uniform distribution

(b) Initialization from WiFi posterior

(c) Initialization from WiFi posterior and occupancy grid

Fig. 2. Different initialization methods for MCLs.

independently learned for each access point from the training dataset, yielding m different path loss models.

A squared exponential kernel was chosen as the kernel function, with free parameters σ_{se}^2 (known as the signal variance), and l_{se} (known as the length-scale). A single kernel is used for all access points, with free parameters, often referred to as hyper-parameters ($\theta_{se} = [\sigma_{se}^2,\ l_{se}]$), also learned from the training dataset.

For making new predictions $z_{*,j}$ at any arbitrary location $\mathbf{x}_*$, we condition the predicted signal strength vector on its location and the training dataset, obtaining

$$p(\mathbf{z}_*|\mathbf{x}_*, \mathbf{X}, \mathbf{Z}) \sim \mathcal{N}(\mathbb{E}[z_{*,j}], \mathrm{var}(z_*)) \tag{3}$$

where,

$$\mathbb{E}[z_{*,j}] = PL_j(\mathbf{x}_*) + \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I}_n)^{-1}(\mathbf{Z} - PL_j(\mathbf{X})) \tag{4}$$

$$\mathrm{var}(z_*) = k_{**} - \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2\mathbf{I}_n)^{-1}\mathbf{k}_*, \tag{5}$$

with $\mathbf{K} = cov(\mathbf{X}, \mathbf{X})$ being the covariance matrix between all training points $\mathbf{X}$, usually called Gram Matrix; $\mathbf{k}_* = cov(\mathbf{X}, \mathbf{x}_*)$ the covariance vector that relates the training points $\mathbf{X}$ and the arbitrary location $\mathbf{x}_*$; $k_{**} = cov(\mathbf{x}_*, \mathbf{x}_*)$ the variance of the location.

The likelihood of an individual access point j for a new measurement $z_{new,j}$ at a location $\mathbf{x}_*$ is then computed as,

$$p(z_{new,j}|\mathbf{x}_*) = \Phi(z_{new,j} - \mathbb{E}[z_{*,j}], \mathrm{var}(z_*)); \tag{6}$$

and the integrated likelihood of all access points, $p(\mathbf{z}_{new}|\mathbf{x}_*)$, as the geometric mean of all individual likelihoods,

$$p(\mathbf{z}_{new}|\mathbf{x}_*) = \left(\prod_{j=1}^{m} p(z_{newj}|\mathbf{x}_*)\right)^{1/m}, \tag{7}$$

which due to heading direction having been assumed no to influence signal strength measurements is equivalent to the sensor model $p(\mathbf{z}|\mathbf{l})$.

III. W-LRF

In this section we describe in detail the two additional components of our approach, our global localization and localization failure detection modules. As well as give the overall algorithm of our whole approach.

A. Global localization

In order to solve the global localization problem, $p(\mathbf{l}_0)$ is often initialized using an uniform distribution over the entire map (see Fig. 2a). Particle initialization is extremely important as the solution may fail to converge to the true pose if too few particles are placed in its vicinity at initialization. This happens as the particle filter implementation of $p(\mathbf{l}_t)$ is based on constant re-sampling of particles at previous times, if no particles are in the vicinity of a high probability area at any given point, these high probability areas will not be populated in next time steps, resulting in $p(\mathbf{l}_t)$ deviating from the true distribution. Therefore, leading to poor accuracy or failure of estimation. While increasing the number of particles ensures better accuracy and convergence, it can not be indiscriminately increased, as it increases convergence times and computation costs. Hence, what is desired is high density of particles around the true location, with as few particles as possible.

Instead of an uniform distribution, $p(\mathbf{l}_0)$ can be initialized by sampling the WiFi models' posterior (see Fig. 2b). This allows for a large number of particles at the true pose vicinity while keeping the number of initial samples relatively low. This initialization can be further improved by also considering the occupancy grid map required for the laser rangefinder perceptual model (see Fig. 2c). One notable compromise of this initialization is that sampling from WiFi models is computationally more expensive than sampling from the uniform distribution.

To sample from the WiFi posterior given a RSS measurement $\mathbf{z}$, $p(\mathbf{x}|\mathbf{z})$, we use a modified adaptive rejection sampling algorithm. In rejection sampling, first an easy to sample proposal distribution $q(\mathbf{x})$ has to be chosen. The main consideration is that given a constant k, $kq(\mathbf{x}) \geq p(\mathbf{x})$. Then samples $\mathbf{x}_s$ are drawn from $q(\mathbf{x})$ as well as random numbers u_s sampled uniformly at random from $[0, kq(\mathbf{x}_s)]$. Finally, if $u_s > p(\mathbf{x}_s)$, the sample is rejected, if not, it is accepted.

For our implementation we discretize the entire environment into $K \times K$ x-y cells. For each cell we compute its weight w_k as the WiFi posterior of its centroid using eq. (7). We employ this discrete function as our proposal

distribution. In order to sample from it we use importance sampling to select one cell and then uniformly at random select a candidate sample $\mathbf{x}_s$ within the cell. When using information from the occupancy grid map, we immediately reject the sample if it is not in an *empty* cell - i.e., a cell that was found to be open space when the map was generated. Finally, if the sample was not discarded, we evaluate the particle's WiFi posterior and use rejection sampling to keep or discard the sample. This process is continued until all desired samples have been obtained. Algorithm 1 outlines our sampling algorithm.

Algorithm 1 Sampling WiFi posteriors

```
Discretize Environment into X_k cells
Compute cell centroids x_k
w_k = p(x_k|z)
i = 0
while i < Nsamples do
    Draw (X_k^(i), w_k^(i)) using importance sampling
    Draw x_s uniformly at random from X_k^(i)
    if x_s is not a free in the grid map then
        reject
    else
        w_s = p(z|x_s)
        Draw u_s uniformly at random from [0, w_k^(i)]
        if u_s > w_s then
            reject
        else
            accept, increase i
```

Using this strategy for sampling allows for much faster computation than using standard rejection sampling. However other sampling methods like Markov Chain Monte Carlo sampling remain to be explored.

After initialization, we continue to use the WiFi in conjunction with LRF for resampling. Once particles have converged into small clusters (usually around a 1m diameter) the particle weights when computed by the WiFi models tend to have the same value, as WiFi localization accuracy is not high enough. Therefore, once these weights converge, we stop using WiFi data for resampling. Using both models allows for faster particle convergence as WiFi data allows the fast elimination of any leftover ambiguity from the initialization, often caused by particles' heading angles.

B. Localization failure detection

Once MCL converges, it becomes highly susceptible to large localization failures, being the extreme case the previously mentioned kidnapped robot problem. Once particles have converged and the localization error occurs, the likelihood of the new position to be covered by any particles is extremely low. A straightforward approach to alleviate the kidnapped robot problem in MCL is the addition of random particles (usually between 1 to 5% of s). This percentage is often made dependent on the particles' weights $\mathbf{w}$, as low $\mathbf{w}$ values indicate wrong pose estimations. The percentage of random particles can be computed based on the average of $\mathbf{w}$ and a fixed threshold. Other approaches include adding particles directly sampled from the posterior of the sensor models instead of random particles [16]; and expanding the region where previous particles were located when particle weights were low [17]. These two approaches detect the occurrence of the kidnapping using the average weight of particles as computed by the laser rangefinder module. Other than average weight, it has also been suggested to use the difference in the average weights as well as the maximum weight of particles [18]. All these metrics are fast to compute; unfortunately, for LRFs in highly symmetric environments, these metrics do not reliably detect the kidnapping.

Instead, the use of WiFi models for detecting localization failure was suggested in our previous work [14]. There, we sampled WiFi posteriors continuously and compared those particles with MCL ones. A large difference between the sets of particles would suggest the occurrence of localization failure. While extremely effective, constant sampling from sensor posteriors posses a high computational toll in the system. In this work we propose a simpler metric, the weight of particles when evaluated using the WiFi models (wifi-weights), as contrary to lrf-weights, wifi-weights do noticeably and reliably vary when the kidnapping occurs due to RSS maps having no symmetries. Tests showing the effectiveness of our approach can be found in Sec. IV-B.

Our W-LRF approach incorporates the ideas presented in this section into a MCL that uses a laser rangefinder. Algorithm 2 shows the overall algorithm of our approach.

Algorithm 2 W-LRF

```
while True do
    Draw p from WiFi posterior using Algorithm 1
    wifi_lrf_update = True
    no_localization_error = True
    while no_localization_error do
        kld_bound_staisfied = False
        while not kld_bound_satisfied do
            draw p^(i) with probability w^(i)
            update p^(i) from the robot motion model
            w_wifi^(i) = wifi_model
            w_lrf^(i) = lrf_model
            if wifi_lrf_update then
                w^(i) = w_wifi^(i) * w_lrf^(i)
            else
                w^(i) = w_lrf^(i)
            kld_bound_satisfied = Check kld bound
        if std(w_wifi) < thr_sd then
            wifi_lrf_update = False
        if mean(w_wifi) < thr_mean then
            no_localization_error = False
```

IV. Experiments

To verify the effectiveness of our approach, experiments were conducted on the third floor of the Engineering Building No. 2 of The University of Tokyo by teleoperating a Pioneer 3DX mobile robot equipped with a laser rangefinder and a laptop. This laptop was used to teleoperate the robot, record all range data from the range finder and WiFi data from its network interface controller. The used laptop was a Panasonic Let's Note CZ-SZ5 laptop which uses a Dual Band Wireless-AC 8260 WNIC and a Core i5 6200U processor. All reported computation times have also been measured using this laptop.

All implementations of our algorithms run using the Robot Operating System (ROS) and RSS data was acquired using *tcpdump* version 4.5.1 as wireless packet analyzer. RSS Data was captured in monitor mode and only beacon frames were recorded.

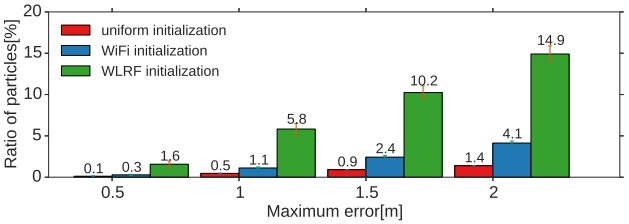
To build the required occupancy grid map and location-signal strength training dataset, the robot was teleoperated around the building. Fast-Slam [19] was used to build the map, and provide the locations at which RSS data was recorded. For testing purposes, the robot was once more teleoperated around the environment, saving all data in time stamped data logs. These time stamped logs allows us test our algorithms thoroughly, using real data and in a real-time manner. Due to the randomized nature of MCL algorithms, this is necessary to guarantee that the results presented are the average expected performances, and not just outliers with good/bad performances; therefore all tests were performed 25 times. We have made all datasets acquired and software developed in this work available online for the community on our web[1].

As we are interested in both the performance of global localization, as well as detection of localization failures, we perform two sets of tests. The first tests are used to assess the quality of our initialization methods as well as its impact on convergence success and speed. The second set of tests are used to assess the detection capabilities of our approach by emulating the kidnapped robot problem.

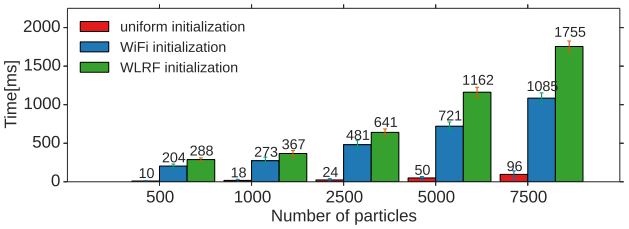
A. Global localization

For global localization, we first start establishing the computation times necessary for sampling, as well as the percentage of particles in the vicinity of the true pose for each of the three possible initializations described in Sec. III-A: uniform initialization, wifi initialization, and wifi initialization with occupancy map information (wlrf initialization). We use the percentage of particles as an indicator of the quality of the initialization as having a large number of particles near the true pose increases the probability of successfully converging to the true pose, while fewer particles allow fast convergence and lowers computational costs for both sampling and the following evaluation of particle weights and resampling process.

[1]http://www.robot.t.u-tokyo.ac.jp/~miyagusuku/software



(a) Percentage of particles placed in the vicinity of the true pose.



(b) Runtime for sampling different number of particles employing the different sampling methods.

Fig. 3. Comparison of sampling methods employed in the global localization module.

Specifically, for all testing points in our test data, we sampled 5000 particles using the different initialization methods. Then, we computed the percentage of particles within different error radius. For this test we ignore the heading angle as all initialization methods sample them at random. Figure 3b shows our results. As it can be observed, wlrf initialization is the one that generates higher density of particles around the true position, with almost 15% of the particles being generated within 2m, compared to 4.1% and 1.4% for wifi and uniform initializations respectively. For all cases wlrf initialization generates more than 3 times the density of wifi initialization and more than 10 times that of uniform initialization.

Figure 3 shows the times required to sample different number of particles using each initialization method. Wlrf initialization is the one requiring the longest time for sampling, between 30 to 60% more time than wifi initialization. While the uniform initialization is by far the fastest, with up to 20 times faster sampling times.

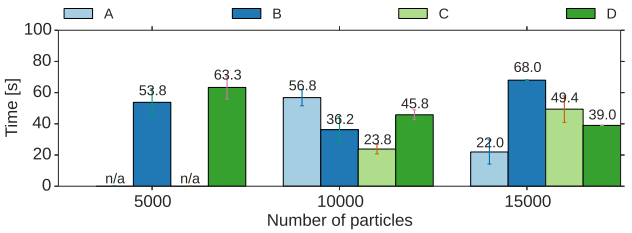
From this result, it may seem that uniform distributions are the best choice, as they generate the most particles per computation time; however, it is important to note that after initialization, MCL needs to continuously evaluate and process all particles. Therefore, a higher concentrated, lower number of particles is still computationally more efficient, regardless of the extra initial computation time for sampling.

To assess the required time for MCL's convergence, we modified the testing data log so the robot starts at different 4 different locations (A,B,C,D) in the environment, see Fig. 5, and obtained the success ratio of localization and the average convergence time when successful for different number of particles. We define the success ratio as the number of successful global localization with respect to the total number of runs, and a successful global localization as one where its particles are within 1m from the true location with at least

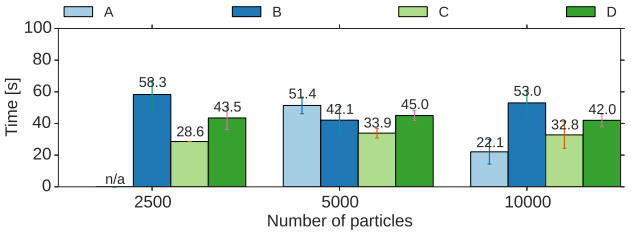
TABLE I

SUCCESS RATIO OF LOCALIZATION WHEN EMPLOYING THE DIFFERENT SAMPLING METHODS.

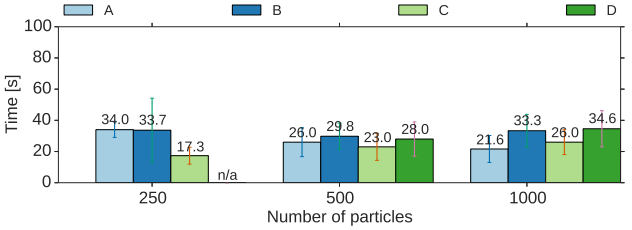
	uniform			wifi			wlrf		
part	5k	10k	15k	1.5k	2.5k	5k	0.25k	0.5k	1k
A	0.0	0.96	1.0	0.00	0.92	1.0	0.50	0.75	1.0
B	1.0	1.00	1.0	0.36	0.60	1.0	0.38	0.62	1.0
C	0.0	0.92	1.0	0.68	0.88	1.0	0.50	0.75	1.0
D	0.8	0.88	1.0	0.36	0.68	1.0	0.00	0.50	1.0



(a) Uniform initialization for 5k,10k and 15k particles.



(b) Wifi initialization for 1.5K, 2.5k and 5k particles.



(c) Wlrf initialization for 250, 500 and 1000 particles.

Fig. 4. Comparison of convergence times when using the different sampling methods.

95% confidence. Table I shows the success ratios for all our tests; while Fig. 4 the average converge times.

As it can be observed from table I, the required number of particles to obtain reliable localization widely vary for each sampling method. As expected, the number of particles required for reliable global localization when using uniform initialization is considerably larger than when using our proposed wlrf initialization (15 against 1 thousands). Furthermore, from Fig. 4 we can observe that less time is required for convergence for all cases. Unexpectedly Wifi initialization still required a large number of particles, around 5 thousand, and convergence times similar to those obtained when using uniform sampling. This is due to most wifi samples not being initialized in empty areas in the grid map, which were then being heavily penalized by the LRF model. This resulted in these particles being eliminated very early in the localization process - equivalent to never having them in

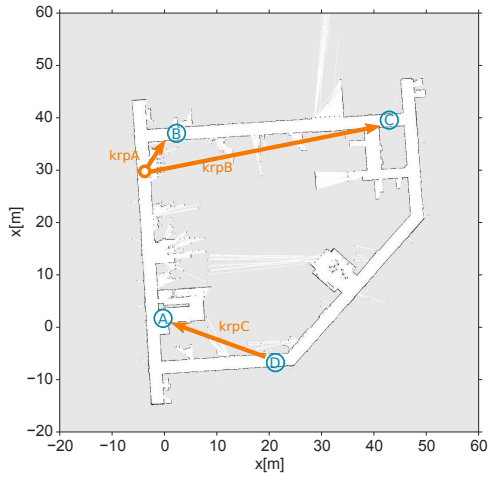


Fig. 5. Floor map of the environment where tests were performed. Locations A, B, C, and D in the environment were used as initial points for testing global localization. While paths krpA, krpB and krpC show the locations of the robot before and after each of the emulated kidnappings.

TABLE II

LOCALIZATION FAILURE DETECTION RESULTS

	p(false positives)				Detection time [s]			
thr	0.05	0.10	0.15	0.2	0.05	0.1	0.15	0.2
krpA	0.0	0.0	0.25	0.42	4.2	2.1	2.7	0.8
krpB	0.0	0.0	0.27	0.54	2.4	2.9	1.7	0.9
krpC	0.0	0.0	0.0	0.0	3.6	2.4	2.8	2.0

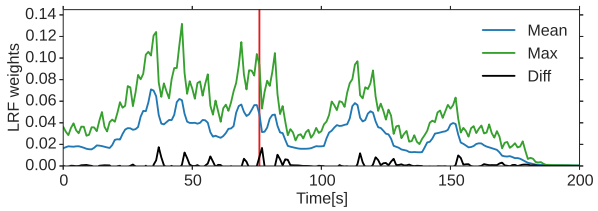
the first place. From these experiments, it becomes obvious that using our proposed initialization should be preferred despite of the higher initial computational requirements.

B. *Localization failure*

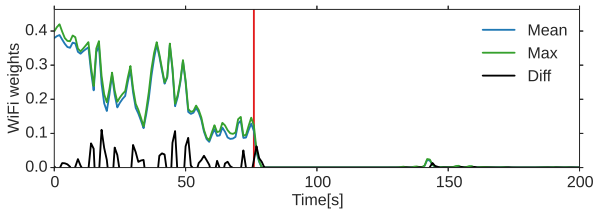
For testing the detection of localization failures, kidnapping of the robot was introduced by modifying the testing data log. This test cases are krpA, krpB and krpC, and can be seen in Fig. 5.

As previously mentioned, typically localization failure is detected using the mean, max or gradient of weights computed by the LRF model. When these metrics go bellow or above certain thresholds, it is considered that a localization failure occurred. Unfortunately in highly symmetric environments with long corridors, as those commonly found in modern offices, these weights do not vary considerably. Figure 6a show particle weights computed by the LRF model (lrf-weigths) in krpA. As it can be observed, neither one of the three metrics noticeably change after kidnapping at time t=73. In fact we could not find any combination of metrics nor thresholds that could reliably detect kidnapping for our three cases.

Our proposed strategy for detecting localization failures uses the particle weights as computed by our WiFi models. As it can be seen in Fig. 6b, for the same test case, wifi-weights notably change after kidnapping, hence can be reliably used for detecting them. For all test cases, the time required for the system to detect the failure as well as the



(a) Particle weights computed by the LRF model



(b) Particle weights computed by the WiFi models

Fig. 6. Particle weights under localization failure (kidnapped robot problem) test case krpA. Red line indicates time at which kidnapping occurred.

probability of false detections was computed for different threshold parameters. The probability of false detection was computed as the percentage of false detections 50 seconds before the actual induced error. Table II shows the results from our tests, where it can be observed that for thresholds of 0.05 and 0.01 no false positives were obtained, while having a fast detection response, between 2 and 4 seconds. Higher threshold values tend to cause a high false positives probability.

V. Conclusions

In this paper we have proposed a novel system which incorporates WiFi-based localization into a typical Monte Carlo localization algorithm that primarily uses laser rangefinders. In our system we employ WiFi data in conjunction with the occupancy grid map of the environment for MCL initialization. By generating high density of particles around the robot's true pose, while keeping particles in other areas to a minimum, the system greatly alleviates laser rangefinder's data association problem - as areas with similar range scans are not populated with particles. Experimental results show that with between 500 and 1000 particles our system achieves convergence, opposed to 5k required when only WiFi data is used and over 15k with a standard MCL approach. We attribute the further improvement of our approach over WiFi only initialization to several wifi samples being initialized in occupied areas when grid map information is not used. In following iterations, these samples are heavily penalized by the LRF model, quickly disappearing. This wastes resources, and is equivalent to initializing the system with fewer samples. Other than this initialization, our system resamples using not only range but also WiFi data, which allows for convergence times to be considerably reduced. In addition, by monitoring particle weights computed using our WiFi models, our system can successfully detect any possible localization failure. To test this characteristic, several tests have been performed where the fully converged system was displaced to an arbitrary location, i.e., the kidnapped robot problem. Notably, opposed to our previous work, the additional computations required for localization failure detection are minimum.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[2] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, pp. 343–349, 1999.

[3] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, "Particle filters for mobile robot localization," in *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 401–428.

[4] T. Rofer and M. Jungel, "Vision-based fast and reactive monte-carlo localization," in *Robotics and Automation (ICRA), 2003 IEEE International Conference on*, vol. 1, 2003, pp. 856–861.

[5] R. Miyagusuku, A. Yamashita, and H. Asama, "Improving gaussian processes based mapping of wireless signals using path loss models," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, 2016, pp. 4610–4615.

[6] J. S. Gutmann, E. Eade, P. Fong, and M. E. Munich, "Vector field slam - localization by learning the spatial variation of continuous signals," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 650–667, June 2012.

[7] P. Bahl and V. N. Padmanabhan, "Radar: an in-building rf-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2000, pp. 775–784 vol.2.

[8] B. Benjamin, G. Erinc, and S. Carpin, "Real-time wifi localization of heterogeneous robot teams using an online random forest," *Autonomous Robots*, vol. 39, no. 2, pp. 155–167, 2015.

[9] A. LaMarca, J. Hightower, I. Smith, and S. Consolvo, "Self-mapping in 802.11 location systems," in *UbiComp 2005: Ubiquitous Computing*. Springer, 2005, pp. 87–104.

[10] B. Ferris, D. Haehnel, and D. Fox, "Gaussian processes for signal strength-based location estimation," in *In Proc. of Robotics Science and Systems*, 2006, pp. 1–8.

[11] J. Biswas and M. Veloso, "Wifi localization and navigation for autonomous indoor mobile robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 4379–4384.

[12] ——, "Multi-sensor mobile robot localization for diverse environments," in *RoboCup 2013: Robot World Cup XVII*. Springer, 2014, pp. 468–479.

[13] S. Ito, F. Endres, M. Kuderer, G. Diego Tipaldi, C. Stachniss, and W. Burgard, "W-RGB-D: floor-plan-based indoor global localization using a depth camera and wifi," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 417–422.

[14] Y. Seow, R. Miyagusuku, A. Yamashita, and H. Asama, "Detecting and solving the kidnapped robot problem using laser range finder and wifi signal," in *Proceedings of the 2017 IEEE International Conference on Real-time Computing and Robotics (RCAR2017)*, July 2017, pp. 303–308.

[15] D. Fox, "Adapting the sample size in particle filters through kld-sampling," *The international Journal of robotics research*, vol. 22, no. 12, pp. 985–1003, 2003.

[16] S. Lenser and M. Veloso, "Sensor resetting localization for poorly modelled mobile robots," in *Robotics and Automation (ICRA), 2000 IEEE International Conference on*, vol. 2, 2000, pp. 1225–1232.

[17] R. Ueda, T. Arai, K. Sakamoto, T. Kikuchi, and S. Kamiya, "Expansion resetting for recovery from fatal error in monte carlo localization-comparison with sensor resetting methods," in *Intelligent Robots and Systems (IROS), 2004 IEEE/RSJ International Conference on*, vol. 3, 2004, pp. 2481–2486.

[18] I. Bukhori, Z. Ismail, and T. Namerikawa, "Detection strategy for kidnapped robot problem in landmark-based map monte carlo localization," in *Robotics and Intelligent Sensors (IRIS), 2015 IEEE International Symposium on*, 2015, pp. 75–80.

[19] S. Zaman, W. Slany, and G. Steinbauer, "Ros-based mapping, localization and autonomous navigation using a pioneer 3-dx robot and their relevant issues," in *Electronics, Communications and Photonics Conference (SIECPC), 2011 Saudi International*, 2011, pp. 1–5.