2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

RSS-Based Relative Localization and Tethering for Moving Robots in Unknown Environments

Stefan Zickler and Manuela Veloso
Computer Science Department, Carnegie Mellon University, Pittsburgh, USA
{szickler,veloso}@cs.cmu.edu

Abstract— **The LANdroids project requires robots to autonomously localize, track, and follow (a task also known as tethering) other robots or humans in an unknown environment with limited sensing abilities. In this paper, we present a localization and tethering approach that relies solely on wireless signal strength and robot odometry without requiring any known reference points in the domain. We introduce a data-driven, probabilistic model that maps received signal strength (RSS) values to real-world distance distributions and embed this model in a grid-based localization algorithm that successfully performs the LANdroids tethering task. We furthermore show, that it is possible to improve localization through the addition of a compass sensor and inter-robot information sharing.**

I. Introduction

The LANdroids project aims to develop an intelligent wireless mesh network consisting of multiple small, low-cost robots [1]. Once deployed, this self-organizing robot network can then be used as a sensing and communication platform.

One particular sub-goal of LANdroids is to support *tethering* functionality. Tethering is the task of a robot localizing, tracking, and following a moving target (either human or robot) in order to provide it with continuous network coverage. Solving the LANdroids tethering problem is very challenging for several reasons:

- Random initialization and lack of domain map.
  In a LANdroids domain, robots are not provided with any prior map of their domain. Additionally, at initialization, all LANdroids robots are placed at random locations with random orientations. Therefore, robots lack a shared reference frame and are unaware of their mutual relative positions and orientations.
- Limited sensing.
  Due to their desired low-cost design, LANdroids robots have very limited sensing abilities, providing only a 2.4GHz wireless radio, odometry feedback, and some reactive sensing (push bump-sensor and a single short-range IR).

To solve the tethering problem, we introduce a data-driven, probabilistic approach that relies on the robot's received signal strength (RSS) as its primary means for localization. Although developed for the LANdroids domain, our approach is very general and should therefore also be considered applicable in other scenarios, such as mobile sensor networks or other multi-robot tasks.

This paper is organized as follows: In Section II we review related work. We then introduce our data-driven probabilistic model for RSS-based ranging in Section III. Section IV presents our grid-based localization algorithm and its integration into the tethering behavior. Finally, we show results of our approach in Section V, followed by concluding remarks and ideas for future work in Section VI.

II. Related Work

Much work on RSS-based localization originates from the realm of sensor networks. Several papers cover the problem of localizing a mobile node in relation to an existing static network of nodes [2]. The RADAR project early on performed WLAN-based localization of a moving user by relying on signal strength readings [3], [4]. Recent approaches have refined this technique by modeling the localization estimate using Bayesian Methods [5]. However, all of these approaches assume some prior sensor network configuration knowledge, such as the locations of static wireless nodes in the domain, or at least an environmental map with pre-recorded observed signal strengths of static access points [6]. Even when attempting to localize nodes in a non-static sensor network, it is typically assumed to have a subset of known reference beacons [7]. Compared to such localization tasks, the LANdroids domain is significantly more challenging because there is absolutely no prior knowledge about transmitter locations and because of the added task of tethering.

The challenge of the RSS-based localization problem arises from the difficulty of extracting range information from signal strength (see also Section III). To overcome these issues, some work makes use of sophisticated hardware to improve the accuracy of the ranging task. Arrays of custom directional antennas can be used to better localize sensor nodes using RSS [8]. Other approaches rely not only on signal strength, but try to infer distance based on the "time-of-flight" signal propagation time between nodes [9]. However, in the LANdroids domain, robot nodes need to rely on single low-cost off-the-shelf omnidirectional antennas without time-based ranging capabilities, thus requiring a localization approach that is able to accommodate readings from such unsophisticated hardware.

Some approaches completely abandon the idea of directly extracting detailed distance information from RSS values, and instead employ a boolean connectivity model [10], [11]. In these approaches, nodes are considered connected if the RSS value lies above a certain threshold, which in turn implies that a receiving node lies within a particular maximum range of the transmitter. Given a boolean connectivity model, it is possible to employ a Monte Carlo localization method to construct a probabilistic localization estimate by letting a mobile node gather RSS samples from varying locations [12]. Our work is distinctive from these approaches because it does not rely on boolean connectivity, but instead introduces a data-driven model that provides a mapping from RSS values to range distributions. Additionally, we not only solve the localization task, but also deal with the LANdroids mobile tethering task.

III. RSS-Based Ranging

Our approach uses the wireless radio's received signal strength indicator (RSSI) to perform ranging. RSSI aims to represent the strength of a received radio signal, but does not deliver values in any physical unit such as mW or dBm. Instead, RSSI provides a unitless measurement in a pre-specified range. For all wireless receivers used in our work, RSSI is expressed as an integer in the range from 0 to 100.

To use RSSI as a means for range estimation, we need to first understand and model the relationship between transmission distance and signal strength. Intuitively, modeling this relationship might seem simple, as one might expect signal strength to decrease (ideally monotonically) as the distance between transmitter and receiver increases. However, attempts to model this signal/range relationship through a simple mapping function (i.e. a polynomial) are likely to fail for any real-world domain because of multiple reasons:

- Environmental Interference
  Any physical structure has the potential to significantly weaken signal strength. RSSI between two nodes can vary greatly depending on the number of walls and other physical structures between them. Additionally, many environments carry sources of active wireless interference, especially in the commonly used frequency bands, such as 2.4GHz. Given the fact that our robot is not provided with a map of its environment a priori, a low signal strength can either indicate a great distance between transmitter and receiver, or simply heavy interference.
- Multipath Interference
  Another reason that makes it extremely difficult to accurately infer distance from RSSI is the phenomenon of *multipath interference* [13]. Multipath interference occurs if multiple instances of the same original signal wave arrive at the receiver after traveling on multiple paths of different length. The addition of these multiple waves can lead to significant interference if they are received out of phase. The typical observed result of multipath interference is the creation of many local signal strength peaks and valleys scattered throughout the domain.

A. Probabilistic, Data-Driven RSS/Distance Model

Due to the difficulty to accurately model the relationship between distance and signal strength analytically, we pursue a data-driven probabilistic approach instead. Rather than attempting to generate a unique mapping from RSSI to distance, our model probabilistically represents the likelihood of a node being at a particular distance, given an observed RSSI reading. Formally, we define the likelihood function

$$L(\texttt{dist}|\texttt{rssi}).$$

To compute this likelihood, we use an explicitly constructed model consisting of a finite discrete set of probability distributions, for the different RSSI integer values 0 to 100:

$$\{D_0, \ldots, D_{100}\}.$$

The purpose of a distribution D_i is to compute the probability p of a receiving node being at a particular distance from the transmitter, that is $p = D_i(\texttt{dist})$. Each probability distribution D_i corresponds to a particular observed RSSI signal strength of value i. We can now fully define L such that

$$L(\texttt{dist}|\texttt{rssi}) = D_{\texttt{rssi}}(\texttt{dist}).$$

Each probability distribution D_i is modeled as a symmetric trapezoidal distribution defined by the tuple $\langle \mu, \tau, \sigma \rangle$ where, as shown in Figure 1, μ is the mean of the distribution, τ is the radius of the trapezoid's uniform center component (where $p = 1$), and σ is the radius of the trapezoid's linear falloff component. Given this tuple, we can now define how D_i computes the probability p given a particular distance $\texttt{dist}$:

$$p = D_i(\texttt{dist}) = \frac{\max(0, \sigma - \max(0, |\texttt{dist} - \mu| - \tau))}{\sigma}.$$

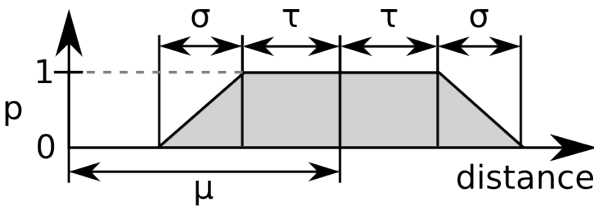


Fig. 1. A symmetric trapezoidal probability distribution.

The trapezoidal model was chosen because it acts as a good approximation of typically observed real world distance distributions. Due to the various interference effects (see Section III), such distributions tend to have a wide and nearly uniform body without strong modal peaks, thus making a trapezoid a much more preferable model over, e.g., a Gaussian distribution.

B. Model Generation

To obtain a usable likelihood function L, it is required to collect a sufficient amount of real-world data to approximate the values of $\langle \mu, \tau, \sigma \rangle$ for each distribution D_i. We do so by collecting pairs of observed RSSI values and their corresponding ground-truth measured distance between transmitter and receiver. In order for the resulting model to be applicable in target domains of arbitrary configurations, this data-set should be as extensive as possible, encompassing the full range of transmitter/receiver distances and varying numbers of walls between them as can be expected for any typical LANdroids domain. Such a data-set can be generated by running multiple robot-nodes in parallel, placing them at randomly selected locations that are carefully measured in some global coordinate frame, and then recording RSS values between all connected pairs. Furthermore, it is possible to let robots drive a fixed distance (depending on odometry accuracy) to automatically obtain additional RSSI samples from varying ground-truth positions.

Once this dataset is obtained, we can extract the parameters for each distribution D_i by computing statistics over all collected data-pairs with an RSSI value of i. We compute the parameters $\langle \mu, \tau, \sigma \rangle$ of a particular distribution D_i from a set of n ground truth distance values $d_1, \ldots, d_n$ that all occurred when an RSSI value of i was recorded. We compute the values as follows:

$$\mu = \frac{\min(d_1, \ldots, d_n) + \max(d_1, \ldots, d_n)}{2}$$

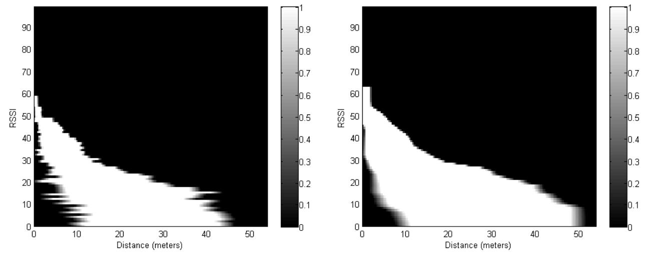
$$\tau = \max\left(\tau_{min}, \frac{\max(d_1, \ldots, d_n) - \min(d_1, \ldots, d_n)}{2}\right)$$

$$\sigma = \max(\sigma_{min}, \operatorname{stddev}(d_1, \ldots, d_n))$$

where τ_{min} and σ_{min} are user-defined constants, used to enforce a certain minimum width of each trapezoidal distribution.

C. Post Processing: Model Smoothing

Even when collecting a large dataset, one should expect the resulting model to be noisy and possibly incomplete. For example, there might exist gaps where no ground truth distance measurements have been observed for a particular RSSI value. Similarly, one might find strong fluctuations in the parameters of neighboring distance distributions. Neither of these effects are desirable as they essentially lead to "overfitting" to an incomplete model. To alleviate these problems, we perform a post-processing step that attempts to smooth out any remaining gaps and inconsistencies in the data model. The smoothing itself is performed through a convolution filter that uses a multi-row sliding window to average the parameters of neighboring distributions. During smoothing, this filter also enforces monotonicity by ensuring that the boundaries (the min and max distance values modeled by D_i) over all trapezoidal distributions D_i do not decrease as i decreases. Figure 2 shows an example of the likelihood model before and after post-processing.



(a) Unsmoothed (b) Smoothed

Fig. 2. Plots of the RSSI/distance likelihood model before (a) and after (b) post-processing. Each horizontal slice represents a trapezoidal distribution D_i for a particular RSSI value i from the vertical axis.

IV. Localization and Tethering

A LANdroids network consists of multiple nodes. A *node* in LANdroids typically refers to a mobile robot, but can also refer to a human who is participating in the network. A node's state s is defined as its 2D position (x, y) and orientation (θ) in its *local coordinate frame*:

$$s = \langle x, y, \theta \rangle.$$

At initialization, each node's local coordinates are set to *the local origin* of $\langle 0, 0, 0 \rangle$. From a global view, nodes are randomly placed and oriented at initialization which implies that no two nodes are likely to share the same local coordinate and orientation frame.

The LANdroids tethering problem is a two-node problem involving a *tracker* node (the node performing the localization and tethering) with state s_a and a *trackee* node (the node that is being localized and followed) with state s_b.

To perform the tethering task, the tracker robot node needs to first establish the relative location of the trackee node that it is supposed to tether to. To establish this location, our approach makes use of continuous measurements of RSSI in combination with the robot's controlled driving and odometry reading abilities.

Our localization approach uses a discrete probability grid to model the position estimates of the trackee node. We define the localization grid G as a set of k cells $\{c_0, \ldots, c_{k-1}\}$, arranged in a 2D grid, w cells wide and h cells high (such that $wh = k$). Each cell c_i contains a tuple $\langle l, p \rangle$ where l is the 2D location of the center of the given cell in the tracker node's local coordinate frame, p is the probability that the trackee robot is located within that given cell (initially, $p = 1/k$ for all cells). Note that, for all of our algorithms we use subscript notation, such as l_i or p_i, to refer to the parameters of the i-th cell. We furthermore use the dot symbol to refer to a member of a data-structure (i.e. $l_i.x$ refers to the x component of the location vector l_i).

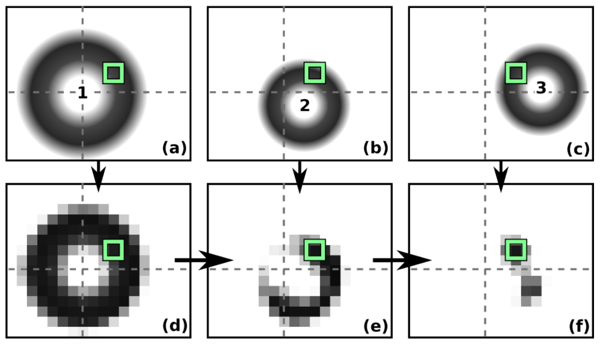


Fig. 3. A visualization of distance distributions from multiple RSSI measurements ((a)-(c)) and the resulting progression of the probability grid ((d)-(f)). Darker color indicates higher probability. Dashed lines indicate the origin in the tracker's coordinate system. The square indicates the trackee's true location.

Figure 3 shows a visual example of how the probability grid is used for localization. In this example, the tracker node takes RSSI measurements from three different locations (annotated 1,2,3). According to the likelihood model L, each measured RSSI value `rss` provides a distance distribution $D_{\texttt{rss}}$, effectively defining a "donut"-shaped probability mass of where the trackee is likely located (see (a)-(c)). Although each independent observation is rather uninformative, integrating the sequence of observations into the probability grid via a Bayesian update function (see (d)-(f)), yields a final distribution that approximates the tracker's true position much more closely (see (f)).

We use the *ProbabilityUpdate* function (see Algorithm 1) to integrate new RSSI observations into the probability grid by performing the Bayesian update step. To perform the Bayesian update, the algorithm multiplies each grid cell's prior probability p_i with its respective result of the likelihood function L (given the cell's distance from the tracker `dist` and the observed RSSI value). The entire probability grid is then normalized, thus resulting in the full posterior probability distribution. Note, that to ensure numeric stability, and to prevent the grid from becoming accidentally irreversibly localized on the wrong target, we introduce a user-defined minimum cell probability ϵ that is enforced during the probability update.

Algorithm 1: ProbabilityUpdate

```
Input: rss-value: rss; minimum cell-probability: $\epsilon$.
for $i \leftarrow 0$ to $k-1$ do
    distance $\leftarrow \sqrt{(l_i.x - s_a.x)^2 + (l_i.y - s_a.y)^2}$;
    $p_i \leftarrow p_i\ L(\text{distance}|\text{rss})$;
    $p_i \leftarrow \max(p_i, \epsilon)$;
norm $\leftarrow \left(\sum_{i=0}^{k-1} p_i\right)$;
for $i \leftarrow 0$ to $k-1$ do
    $p_i \leftarrow p_i/\text{norm}$;
```

To retrieve the current localization estimate from the grid, we introduce the *GetLocation* function (see Algorithm 2) that is able to retrieve the grid's probability peak with sub-cell-width precision. This algorithm first finds the cell with the peak probability $c_{\text{max\_idx}}$ and then computes a location offset vector that incorporates the distances and current probabilities of all 8-connected cell neighbors through linear interpolation. Adjusting the peak cell's center location with this offset vector provides us with the final 2D location estimate final_loc.

Algorithm 2: GetLocation

```
max_prob $\leftarrow \max(p_0, \dots, p_{k-1})$;
max_idx $\leftarrow \underset{i \in (0,\dots,k-1)}{\arg\max}\ (p_i)$;
final_offset $\leftarrow (0, 0)$;
vsum $\leftarrow 0$;
foreach neighboring cell $c_j$ of $c_{\text{max\_idx}}$ do
    v $\leftarrow (p_j/\text{max\_prob})$;
    vsum $\leftarrow$ vsum + v;
    final_offset $\leftarrow$ final_offset + (v $|l_j - l_{\text{max\_idx}}|$);
final_offset $\leftarrow$ final_offset/vsum;
final_loc $\leftarrow l_{\text{max\_idx}}$ + final_offset;
return final_loc;
```

Besides finding the localization estimate, it is also important to model its confidence. For this purpose, the *GetUncertainty* function (see Algorithm 3) computes the location uncertainty, acting as a global indicator for how dispersed the grid's probability mass is in relation to its peak.

Algorithm 3: GetUncertainty

```
max_idx $\leftarrow \underset{i \in (0,\dots,k-1)}{\arg\max}\ (p_i)$;
return $\sqrt{\sum_{i=0}^{k-1} p_i \left((l_i.x - l_{\text{max\_idx}}.x)^2 + (l_i.y - l_{\text{max\_idx}}.y)^2\right)}$;
```

Given the introduced algorithms, we now define the core localization and tracking behavior *TetherBehavior* (see Algorithm 4) that runs on the tracker robot. The algorithm begins by assuming that it has no initial knowledge of the trackee's orientation frame nor its location. Provided that a wireless connection between the tracker and trackee exists, the trackee is able to send its odometry pose (in its own local - and possibly unknown - orientation frame) that is stored in s_b. We then enter the main loop. Here, s_b' is the pose of the trackee from the previous loop iteration, whereas s_b is the current trackee pose. If there is no knowledge of the trackee's orientation frame (framesAreSynched = *false*), then we model the motion of the trackee by applying a circular Gaussian blur on the probability grid with a radius of the odometry distance driven by the trackee plus some additional user-defined odometry uncertainty constant γ. If however, the orientation frames between tracker and trackee have been synchronized (i.e., through the use of an added compass), then we can model the motion of the trackee more deterministically by shifting the entire probability grid by the relative translation component of the trackee motion since the last update. Odometry uncertainty is again modeled explicitly by blurring the probability distribution by a radius of γ.

Algorithm 4: TetherBehavior (Tracker Robot)

```
framesAreSynched $\leftarrow$ false;
$s_b \leftarrow$ ReceiveTrackeePose();
repeat
    $s_b' \leftarrow s_b$;
    $s_b \leftarrow$ ReceiveTrackeePose();
    if framesAreSynched then
        ShiftGrid($s_b.x - s_b'.x$, $s_b.y - s_b'.y$);
        BlurGrid($\gamma$);
    else
        BlurGrid($\sqrt{(s_b.x - s_b'.x)^2 + (s_b.y - s_b'.y)^2} + \gamma$);
    loc $\leftarrow$ GetLocation();
    if GetUncertainty() $< v$ then
        DriveToward(loc);
        if haveCompass & (dist($s_a$,loc) $< \delta$) then
            SyncOrientationFrames();
            framesAreSynched $\leftarrow$ true;
    else
        ExploreGrid();
    ProbabilityUpdate(MeasureRSSI(),$\epsilon$);
until aborted ;
```

After updating the grid with the trackee's latest odometry, the algorithm then extracts the current peak trackee location estimate (loc) and selects a particular robot motion behavior to execute. If the trackee is not sufficiently localized then the robot runs a simple exploration behavior, `ExploreGrid()`, that will continuously drive the robot through the grid to collect more RSSI localization samples. If the trackee is considered sufficiently localized (that is, the uncertainty value is below some threshold υ) then the robot drives toward this location, essentially attempting to stay tethered to the trackee.

If both the tracker and trackee are equipped with an orientation sensor (e.g., compass) then the algorithm attempts to synchronize their two orientation frames. To perform this synchronization step, both nodes need to be sufficiently close to each other to minimize any local interference effects of each compass sensor's magnetometer.

After executing its motions, the algorithm updates the probability grid with a new RSSI observation and then continues to repeat the loop, unless aborted by a user.

V. Experimental Results

We tested our approach both in simulation and on real-world LANdroids robots. The algorithm was implemented in the Intelligent Automation, Inc. (IAI) Distributed Control Framework (DCF) [14]. DCF provides a complete physical agent control infrastructure, allowing the integration of the robot behaviors and its related tasks such as inter-robot messaging and RSSI readout. Additionally, DCF comes with a complete simulation framework that includes simulation of RSS interference effects based on walls, as well as simulation of odometry uncertainty and physical collisions.

Quantitatively evaluating our algorithm's localization precision requires ground truth position data for both the tracker and trackee, obtained while the algorithm is running. This is best achieved in DCF's simulated execution environment as it provides global access to the true positions of the robots that can then be compared with the localization estimate obtained by our algorithm.

We performed simulated localization experiments with and without a global orientation sensor on the robots. For our experiments, the trackee was set up to remain stationary until initially localized by the tracker. Once localized, the trackee then starts to move, thus requiring the tracker to continuously re-localize and follow it. Figure 4 shows the results obtained from simulation without the use of a orientation sensor. In this example, the trackee is considered sufficiently localized at approximately 50 seconds, after which it starts to move and the tracker is trying to follow it. Note, that while the true distance between tracker and trackee typically only reaches about 3m, the localization error often peaks highly at about 7m. Figure 5 shows results with the use of orientation sensing. Enabling this synchronization of orientation frames dramatically improves the localization accuracy of the algorithm, now typically remaining in range of 1m during the mobile tracking stage.

The reason for this improvement becomes clear when we look at the probability grid for both scenarios. Without

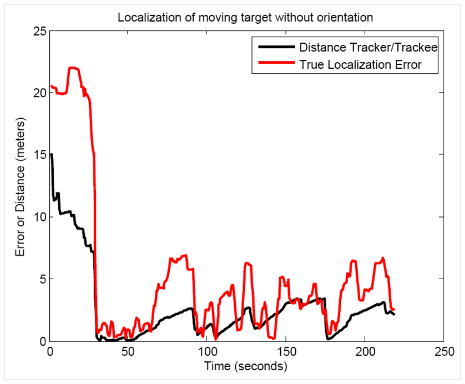


Fig. 4. Localization without orientation sensing.

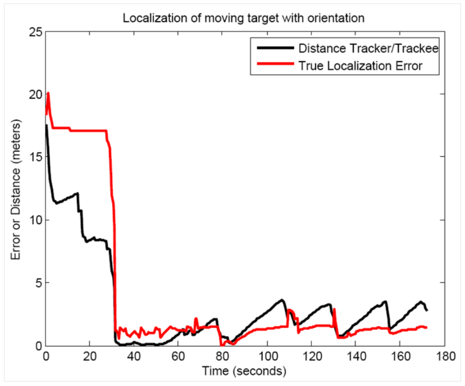


Fig. 5. Localization with orientation sensing.

synchronized orientation frames, the odometry information that is shared by the trackee is relatively uninformative, thus requiring us to apply the `BlurGrid()` function repeatedly, leading to a very uncertain localization (see Figure 6 (a)-(c)). With orientation sensing however, the peak of the grid's probability distribution is well maintained as the trackee's odometry information can be accurately applied to the probability grid (see Figure 6 (d)-(f)).

A simplified real-world tethering demonstration of our algorithm is shown in the video accompanying this paper (also available at `http://www.cs.cmu.edu/~szickler/landroids/`). Unlike the simulated experiments, this real-world demonstration was limited to a large single-room environment to help overcome some remaining limitations in the real-world implementation of our approach. One major such limitation is that `ExploreGrid` and `DriveToward` are mostly reactive and currently do not perform any sophisticated mapping of walls and other obstacles. Due to that fact, it is extremely challenging for our robot to drive to an arbitrary grid-cell without accumulating an excessive odometry-error or getting physically "stuck" due to some real-world obstacle. However, our single-room testing environment still contained large variances of RSS (due to multipath) and required the robot to reactively navigate to reach all steps of the tethering sequence.

Figure 7 shows the two LANdroids robots used for the real-world tethering test of our algorithm. For optional

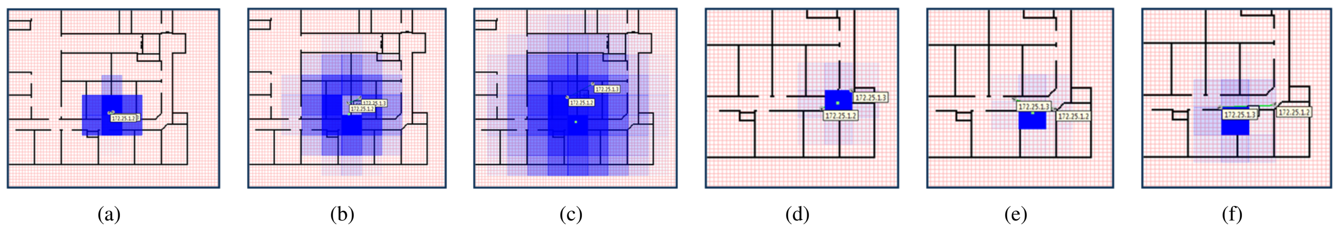


Fig. 6. Two time series of tracking simulation screenshots of a moving target without ((a)-(c)) and with ((d)-(f)) synchronized orientation information. The node (very small grey circle) with the IP 172.25.1.2 is the tracker, the other node is the trackee. Blue shading indicates grid cell probabilities. Cell width is 2m. A small green dot indicates the interpolated predicted position of the trackee.

orientation sensing, each robot carries a Hitachi HM55B compass module. Using the likelihood model shown in Figure 2 (b) and a grid of 400 cells, each with a width of 1.6m, the algorithm performed efficiently, even on the robot's limited computational hardware. In the real-world test, localization of a static target took approximately 1 minute. The magnitudes of visible localization errors encountered during real-world tethering were similar to the ones obtained in simulation, resulting in a successful tracking and following behavior.

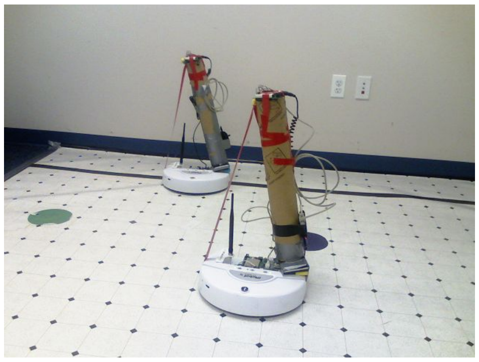


Fig. 7. Two LANdroids robots with added compasses (mounted on tall cardboard pipes to reduce magnetic interference from ground and robot).

VI. Conclusion

We introduced a data-driven, probabilistic model for mapping RSSI signal strength values to range distributions. We presented algorithms that employ this model to perform RSS-based localization and tethering between two mobile nodes that are initialized without a shared coordinate frame and that are limited in their sensing capabilities. We introduced an optional extension to the algorithm allowing the synchronization of the nodes' orientation frames to improve localization accuracy. Finally, we tested our approach, both in simulation and on real-world hardware. For future work, it would be interesting to construct richer behaviors that combine SLAM-style mapping techniques with the localization and tethering work presented in this paper. Furthermore, extending our approach to more than two nodes (multiple trackers and/or trackees) is an interesting research problem.

Acknowledgements

This research was partly sponsored by Intelligent Automation, Inc. (IAI) under subcontract no. 654-1, and by United States Department of the Interior under Grant No. NBCH-1040007. The views and conclusions contained in this document are those of the authors only.

We would like to thank IAI for performing parts of the data collection tasks and for providing access and support to the DCF framework and robot hardware.

References

[1] M. McClure, D. Corbett, and D. Gage, "The DARPA LANdroids program," in *Proceedings of SPIE*, vol. 7332, 2009, p. 73320A.

[2] N. Patwari, J. Ash, S. Kyperountas, A. Hero Iii, R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, 2005.

[3] P. Bahl and V. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *IEEE infocom*, vol. 2, 2000, pp. 775–784.

[4] P. Bahl, V. Padmanabhan, and A. Balachandran, "Enhancements to the RADAR user location and tracking system," *Microsoft Research*, 2000.

[5] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki, "Practical robust localization over large-scale 802.11 wireless networks," in *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2004, pp. 70–84.

[6] J. Biswas and M. Veloso, "WiFi Localization and Navigation for Autonomous Indoor Mobile Robots," in *IEEE International Conference on Robotics and Automation*, 2010.

[7] P. Pathirana, N. Bulusu, A. Savkin, and S. Jha, "Node localization using mobile robots in delay-tolerant sensor networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 3, pp. 285–296, 2005.

[8] J. Ash and L. Potter, "Sensor network localization via received signal strength measurements with directional antennas," in *Proceedings of the 2004 Allerton Conference on Communication, Control, and Computing*, 2004, pp. 1861–1870.

[9] S. Lanzisera, D. Lin, and K. Pister, "RF Time of Flight Ranging for Wireless Sensor Network Localization," *Workshop on Intelligent Solutions in Embedded Systems (WISES), June 2006*.

[10] G. Giorgetti, S. Gupta, and G. Manes, "Optimal RSS threshold selection in connectivity-based localization schemes," in *Proceedings of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems*. ACM New York, NY, USA, 2008, pp. 220–228.

[11] G. Giorgetti, S. K. Gupta, and G. Manes, "Localization using signal strength: to range or not to range?" in *MELT '08: Proceedings of the first ACM international workshop on Mobile entity localization and tracking in GPS-less environments*. New York, NY, USA: ACM, 2008, pp. 91–96.

[12] L. Hu and D. Evans, "Localization for mobile sensor networks," in *Proceedings of the 10th annual international conference on Mobile computing and networking*. ACM New York, NY, USA, 2004, pp. 45–57.

[13] W. C. Jakes, *Microwave Mobile Communications*, 2nd ed. Wiley-IEEE Press, 1994, ch. 1, pp. 11–78.

[14] V. Manikonda, P. Ranjan, and Z. Kulis, "A Mixed Initiative Controller and Testbed for Human Robot Teams in Tactical Operations," *AAAI Fall Symposium, Technical Report FS-07-06*, 2007.