Tracking a moving object with binary sensors

Javed Aslam, Zack Butler, Florin Constantin,
Valentino Crespi, George Cybenko, Daniela Rus

Abstract

In this paper we examine the role of very simple and noisy sensors for the tracking problem. We propose a binary sensor model, where each sensor's value is converted reliably to one bit of information only: whether the object is moving toward the sensor or away from the sensor. We show that a network of binary sensors has geometric properties that can be used to develop a solution for tracking with binary sensors and present resulting algorithms and simulation experiments. We develop a particle filtering style algorithm for target tracking using such minimalist sensors. Our extensive simulations show low error that decreases with sensor density.

1 Introduction

Sensor networks are systems of many small and simple devices deployed over an area in an attempt to sense and monitor events of interest or track people or objects as they move through the area. In general, the sensors used (both the sensor itself as well as any associated computing) are very simple so that their cost remains low. Different sensing modalities including temperature, sound, light and seismic vibrations may be used in such a system depending on the targets of interest.

For several of these sensing modalities, the sensor may generate as little as one bit of information at each point in time. For example, if the sensors are obtaining sound levels, instead of using the absolute sound level (which may cause confusion between loud near objects and quieter close objects), the sensor may simply report whether the sound is getting louder or quieter. Similarly for the seismic sensor, an increase or decrease in intensity can be used. In these systems, using a single bit of information allows for inexpensive sensing as well as minimal communication. This minimalist approach to extracting information from sensor networks leads to a binary model of sensor networks.

In this paper we investigate the minimalist binary model for sensor networks in the context of a tracking application. The binary model assumption is that each sensor can detect one bit of information: whether an object is approaching it or moving away from it. We analyze the binary model assumption in the context of a tracking application and show that for each snapshot of sensor values, the convex hull of the sensors that detect the tracked object is approaching does not intersect the convex hull of the sensors that detect the tracked object is moving away. Furthermore we characterize analytically the constraints imposed by the current location approximation of the object with respect to possible future locations. Based on this result, we derive an algorithm for estimating the trajectory of an object over time, using binary sensor networks. The trajectory of the object is approximated in a discrete fashion. The resolution of the approximation is given by the spacing between the sensor nodes. Finally, we present data from extensive simulations for tracking with binary sensors.

Our algorithm has the flavor of particle filtering and makes two assumptions. First, the sensors across a region can sense the target approaching or moving away. The range of the sensors defines the size of this region which is where the active computation of the sensor network takes place (although the sensor network may extend over a larger area.) The second assumption is the the bit of information from each sensor is available in a centralized repository for processing. This assumption can be addressed by using a simple broadcast protocol in which nodes whose can sense the target send their id and data bit to a base station for processing. Because the data is a single bit (rather than a complex image taken by a camera) sending this information to the base station is feasible. Our proposed approach is most practical for application where the target velocity is slower than the data flow in the network, so that each bit can actually provide information content about the target. However, since the accuracy of our trajectory computation depends on the number of data points, the predictions are not affected by the velocity of the target relative to the speed of communication.

2 Related work

Target tracking is concerned with approximating the trajectory of one or more moving objects based on some partial information, usually provided by sensors. Target tracking is necessary in various domains such as computer vision [3], signal processing and military applications. A typical example is the problem of finding the trajectory of a vehicle by bearings measurement, which is a technique used by radars. Work in robotics has also considered tracking targets from moving platforms [6].

Several methods for tracking have been proposed. This includes Kalman filter approaches or discretization approaches over the configuration space. A recent method that shows great promise is particle filtering, which is a technique introduced in the field of Monte Carlo simulations. The main idea of particle filtering is to discretize the probability distribution of the object's position rather than the position space. This is achieved by keeping multiple copies (called "particles") of the object each of which having associated a weight. With every action (usually a sensor reading) a new set of particles is created from the current one and the weights are updated. Any function about the object is then obtained as the weighted sum of the function values at each particle. The seminal paper in this domain is [5], which states the basic algorithm and properties. Since then many papers have addressed this topic; among the most important are the variance reduction scheme([4]) and the auxiliary particle filter([7]). A survey of theoretical results concerning the convergence of particle filter methods can be found in [2].

Probabilistic methods have also been used in robotics for simultaneous localization and mapping (SLAM), in which the robot attempts to track itself using the sensed position of several landmarks. For example, in [1], particle filter techniques were used for localization only when the traditional Kalman filter technique had failed. These algorithms typically assume range and bearing information between the landmarks and tracked vehicle, unlike the very simple sensors considered here.

We are inspired by this previous work and use the particle filtering approach in the context of the binary sensor model.

3 The binary sensor tracking model

In the binary sensor model, each sensor can detect whether the object is approaching (we will call such a sensor a *plus sensor*) or moving away (we will call such a sensor a *minus sensor*). We assume that the sensor range is such that multiple sensors can detect this information and forward the bit to a base station. We call this the active region of the sensor network.

Because the data is simple and consists of one bit only, this assumption can be met through a protocol in which the active sensors forward their id and data bit. The sensor may be noisy and use thresholding and hysteresis to detect movement and compute the direction bit. The active region of the sensor network may change over time, but since we assume that only the active sensors report data, the computations are done relative to those sensors only. We assume that the base station knows the location of each sensor. Without loss of generality, we assume from now on that all the sensors can sense the object movement over the same space.

In this section we characterize the geometry of the plus sensors and minus sensors instantaneously first, and then over time, using history information. We then relate this characterization to constraints on the trajectory of the object they sense, which will lead to the tracking algorithm developed in the next section.

The tracking problem can be formulated as follows. Suppose a set of m binary sensors $S = \{S_1, S_2, \dots, S_m\}$ are deployed within a bounded $2D$ area. Assume now that an object U is moving inside the area along a curve Γ and let $X(t)$ be one of its parametric representations. Finally, let the sensors sample the environment at regular intervals of time, thereby producing a sequence of binary m-vectors $s \in \{-1, 1\}^m$ (with $s_i^{(j)} = +1/-1$ meaning U is approaching/going away from sensor i at time t_j). Then we would like to provide an estimate of the trajectory X of U for the given placement of the sensors.

3.1 The Instantaneous Sensor Network Geometry

Consider a single sample $s \in \{-1, 1\}^m$ of data, produced at time t. We would like to determine sufficient and necessary conditions for the location X of the target and the direction of its movement $V = X'$.

The key result reported as Theorem 2 shows that the location of the tracked object is outside the convex hull of the plus sensors and also outside the convex hull of the minus sensors. We first show in Lemma 1 an important characteristic of the plus and minus sensors relative to the instantaneous velocity and position of the object.

Lemma 1 *Let i and j be two arbitrary sensors located at positions S_i and S_j and providing opposite information about U at time t. Without loss of generality, let $s_i^{(t)} = +1$ and $s_j^{(t)} = -1$ (object U is shortening its distance from sensor i and increasing its distance from sensor j). Then it must be the case that*

$$S_j \cdot V(t) < X(t) \cdot V(t) < S_i \cdot V(t) \ ,$$

where $\cdot$ denotes the scalar product in $\mathbf{R}^2$.

Proof. Consider the situation as depicted in Fig. 1. Since U is going away from sensor S_j then it must be $\alpha > \pi/2$. Analogously, since U is approaching sensor S_i, it must also be $\beta < \pi/2$. These two conditions translate into

$$(S_j - X) \cdot d\mathbf{l} < 0 \ \text{ and } \ (S_i - X) \cdot d\mathbf{l} > 0 \ ,$$

or in integral form

$$\int_\Gamma (S_j - X) \cdot d\mathbf{l} \quad \text{strictly decreasing,}$$

and

$$\int_\Gamma (S_i - X) \cdot d\mathbf{l} \quad \text{strictly increasing.}$$

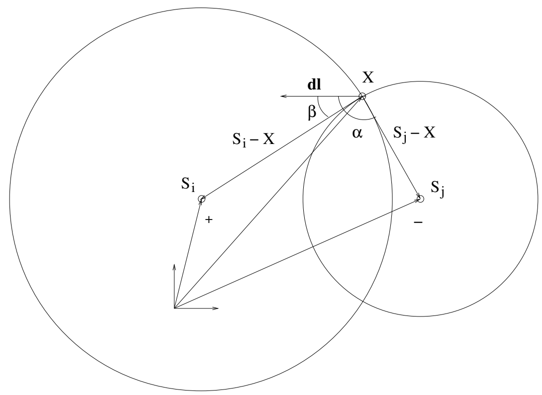

Figure 1: Necessary and sufficient conditions on X.

Replacing $d\mathbf{l} = X'(t)dt$ our conditions become

$$\int_0^t (S_j - X(t)) \cdot X'(t)dt \quad \text{strictly decreasing,}$$

and

$$\int_0^t (S_i - X(t)) \cdot X'(t)dt \quad \text{strictly increasing.}$$

But this amounts to saying that

$$(S_i - X(t)) \cdot X'(t) > 0 \quad \text{and} \quad (S_j - X(t)) \cdot X'(t) < 0 \ ,$$

from which the claim follows. □

An immediate corollary of this lemma is the following condition for the feasibility of a pair (X, V):

$$\max_i \{S_i \cdot V \mid s_i = -1\} < X \cdot V < \min_i \{S_i \cdot V \mid s_i = +1\}.$$

This velocity constraint can be used to derive a useful sensor detection separation result that will further result in object trajectory constraints.

Theorem 2 *Let $s \in \{+1, -1\}^m$ be a sample of the sensor values at a time t. Let $A = \{S_i \mid s_i = +1\}$ and $B = \{S_j \mid s_j = -1\}$ and $C(A)$ and $C(B)$ their convex hulls. Then: $C(A) \cap C(B) = \emptyset$. Furthermore, $X(t) \notin C(A) \cup C(B)$.*

Proof. Assume by contradiction that the first part of the claim is false. Then $C(A) \cap C(B) \neq \emptyset$. This implies that there exists at least one sensor $u \in B$ whose position S_u falls inside $C(A)$. So S_u must be a convex combination of the vertices $\mathbf{a}_j$ of $C(A)$: $S_u = \sum_j \alpha_j \mathbf{a}_j$, with $\alpha_j \geq 0$, $\sum_j \alpha_j = 1$. Now, since $s_u = -1$, by Lemma 1 we must have:

$$\left(\sum_j \alpha_j \mathbf{a}_j \right) \cdot V(t) = \sum_j \alpha_j (\mathbf{a}_j \cdot V(t)) < X(t) \cdot V(t) \ .$$

On the other hand it must also be that

$$\sum_j \alpha_j \mathbf{a}_j \cdot V(t) \geq \sum_j \alpha_j \min_i \{\mathbf{a}_i \cdot V(t)\} \geq \mathbf{a}_{i_0} \cdot V(t) > X(t) \cdot V(t) \ ,$$

which is contradictory. To show the second part of the claim, assume, wlog, by contradiction that $X(t) \in C(A)$. So, as before, $X(t)$ can be expressed as a convex combination of the vertices in $C(A)$: $X(t) = \sum_j \alpha_j \, \mathbf{a}_j$ and by Lemma 1 it must be

$$X(t) \cdot V(t) < \min_j \{\mathbf{a}_j \cdot V(t)\}$$

or by substituting the convex combination

$$\sum_j \alpha_j \mathbf{a}_j \cdot V(t) < \min_j \{\mathbf{a}_j \cdot V(t)\},$$

which is again contradictory. □

Theorem 2 provides a coarse approximation of the location of the tracked object, namely that it has to be outside the minus sensors' and plus sensors' convex hulls. The approximation can be further refined using the following result.

Theorem 3 *Let $s \in \{+1, -1\}^m$ be a sample of the sensors values at a certain time t. Let $A = \{S_i \mid s_i = +1\} \neq \emptyset$, $B = \{S_i \mid s_i = -1\} \neq \emptyset$ and $C(A)$, $C(B)$ their respective convex hulls. Then the normal $\vec{N}$ to the velocity separates $C(A)$ and $C(B)$ and V points to the "+" convex hull.*

Proof. We can suppose modulo a translation of the plane that the current location X of the object is $X = (0, 0)$. Let m be the slope of the velocity and let $\vec{V} = (m \cdot v, v)$ where $v \in \mathbf{R}$. Then the equation of $\vec{N}$ is: $y = -\frac{1}{m} \cdot x$

Let $S^+ = (a^+, b^+)$ an arbitrary "plus" sensor and $S^- = (a^-, b^-)$ an arbitrary "minus" sensor. Then we have to show that

$$\left(\frac{a^+}{m} + b^+\right) \cdot \left(\frac{a^-}{m} + b^-\right) < 0$$

i.e., every "plus" and "minus" sensor lie on different half-planes with respect to $\vec{N}$. What sensors report can be translated as $(S^+ - X) \cdot V > 0$ or $a^+ \cdot v + b^+ \cdot m \cdot v > 0$ and respectively $(S^- - X) \cdot V < 0$ or $a^- \cdot v + b^- \cdot m \cdot v < 0$. By multiplying these relations we get that

$$\left(a^- \cdot v + b^- \cdot m \cdot v\right) \cdot \left(a^+ \cdot v + b^+ \cdot m \cdot v\right) < 0$$

and, by factoring each parenthesis by $m \cdot v$,

$$m^2 \cdot v^2 \cdot \left(\frac{a^-}{m} + b^-\right) \cdot \left(\frac{a^+}{m} + b^+\right) < 0$$

and the claim follows. For the rest part of the claim, note that V points to the "plus" convex hull if and only if $S_+ \cdot V > 0$ or $(a^+, b^+) \cdot (v, m \cdot v) > 0$ or further $a^+ \cdot v + b^+ \cdot m \cdot v > 0$, which is what the sensors read. □

Figure 2 shows the intuition behind the constraints computed based on the sensor geometry. The trajectory of the object is between the convex hull of the plus sensors and the convex hull of the minus sensors. History information accumulated over time can be used to identify the direction and position of the object within this region.

3.2 Linear Programming Perspective

In Section 3.1 we showed some instantaneous analytical properties of trajectories tracked with binary sensors. The proofs presented in this section are intuitive but not constructive. In this section we show how the tracking problem can be formulated constructively in an equivalent fashion using linear programming.

Let (x_0, y_0) be the current position of the tracked object and m_0 the slope of the normal to its velocity. Then the line of slope m_0 passing through x_0 (i.e., the normal to velocity) separates the convex hulls of the "plus" and "minus" sensors. Moreover the velocity points toward the "plus" convex hull.

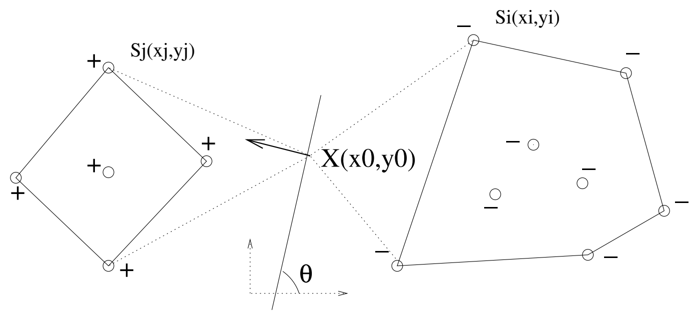

Figure 2: This figure shows the intuition behind the natural constraints on the velocity of the tracked object that are grounded in the convex hull separation result.

Let S_i and S_j be, respectively, sensors with information $-$, $+$. The linear program for the tracking problem an be defined as:

- $-\infty < m_0 < 0$
$$y_i - y_0 \geq m_0 \cdot (x_i - x_0) \qquad y_j - y_0 \leq m_0 \cdot (x_j - x_0)$$
- $m_0 = 0$
$$\max y_i \leq y_0 \leq \min y_j$$
- $\infty > m_0 > 0$
$$y_i - y_0 \leq m_0 \cdot (x_i - x_0) \qquad y_j - y_0 \geq m_0 \cdot (x_j - x_0)$$
- $m_0 = 0$
$$\max x_j \leq x_0 \leq \min x_i$$

The above inequalities can be translated into linear inequalities by introducing a new variable $\mu_0 = m_0 \cdot x_0$. If m_0(the slope) is given then these cases can be reduced to case $m_0 = 0$ by a rotation of angle $-\theta$ where $m_0 = \tan\theta$.

Case $m_0 = 0$ is very convenient because of its simplicity. The domain for y_0 becomes an interval, the boundaries for x_0 being given by the bounded area between the convex hulls.

3.3 Incorporating History

We now extend the instantaneous characterization of the tracked object over time, using history. Consider Figure 3. Intuitively, future positions of the object have to lie inside all the circles whose center is located at a plus sensor and outside all circles whose center is located at a minus sensor, where the radius associated with each sensor S is $d(S, X)$ where X is the previous object location (by $d(A, B)$ we will denote the distance between points A and B). This observation can be formalized as follows.

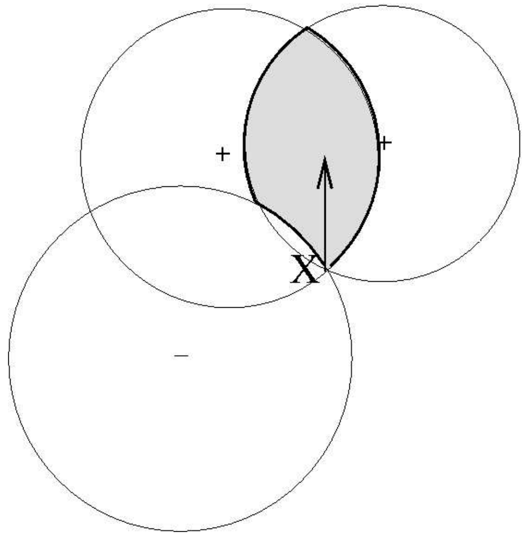

Figure 3: Limitation for future position given previous position

– **Proposition 4** *Let t_0 be a certain time and $t_1 > t_0$ such that sensors S^- and S^+ report – and respectively + at all times $t, \forall t_0 < t < t_1$. Then $\forall t_0 < t < t_1$*

$$d(X(t), S^-) \geq d(X(t_0), S^-) \tag{1}$$
$$d(X(t), S^+) \leq d(X(t_0), S^+) \tag{2}$$

Proof. We prove the claim only for the minus sensor. The other inequality follows by duality. Let

$$\begin{aligned} f(t) &= d(X(t), S^-) - d(X(t_0), S^-) \\ &= (X(t) - S^-) \cdot (X(t) - S^-) - (X(t_0) - S^-) \cdot (X(t_0) - S^-) \end{aligned}$$

We have that $f(t_0) = 0$ and $f'(t) = 2 \cdot (X(t) - S^-) \cdot X'(t) \geq 0$ because S^- reports – at any time t between t_0 and t_1, which means that f is nondecreasing. Since $f(t_0) = 0$, it also follows that $f(t) \geq 0 \, \forall t_0 \leq t \leq t_1$. □

4 Tracking with a Binary Sensor Network

Section 3 gives constraints on the movement of the targeted object, *TO ADD something here: both on ... and on ....* By also assuming that the object's trajectory lies inside the convex hull of all sensors, a tracking algorithm can be developed. Next subsections describe this algorithm and its limitations.

4.1 The Tracking Algorithm

We derive a solution for tracking with binary sensors using the constraints in Section 3 to obtain an algorithm with the flavor of particle filtering. The key idea of the particle filtering method is to represent the location density function by a set of random points (or particles) and to compute an estimation of the true location based on these samples and weights.

Algorithm 1 is a variant of the basic particle filter algorithm. Rather than keeping an equally weighted sample set (as [5] proposes), we use the idea in [4] where each particle has its own weight. The algorithm keeps at each step a set of particles(or possible positions) with weights updated according to the probability of going from the location at time $k-1$ (denoted by x_j^{k-1}) to the location at time k (denoted by x_j^k). This probability is approximated by $\hat{p}(y_k|x_j^k)$. The first particle set is created by drawing N independent particles outside the convex hulls of the "plus" and "minus" sensors at the time of the first sensor reading. Then, with each sensor reading, a new set of particles is created as follows:

1. a previous position is chosen according to the "old" weights
2. a possible successor is chosen for this position
3. if this successor respects acceptance criterion (which is problem-specific and will be described in Subsection 4.2 , add it to the set of new particles and compute its weight . If not, start over.

The above sequence of steps is repeated until N new particles have been generated. The last step is to normalize the weights so they sum up to 1.

Algorithm 1 Particle Filter Algorithm

Initialization: A set of particles $(x_j^1, w_j^1 = 1/N)$ for $j = 1, \ldots, N$
$k = 1$
while y_k (sensor readings) $\neq \emptyset$ (sensors still active) **do**
  $k = k + 1$ **repeat**
  choose $j \sim multinomial(w_1^{k-1}, \ldots, w_N^{k-1})$
  take $x_j^k = \hat{f}_k(x_j^{k-1}, y_k)$
  if x_j^k respects "goodness" criterion **then**
   accept it as a new particle
  end if
 until N new particles have been generated
 for $j = 1 : N$ **do**
  $w_j^k = w_j^{k-1} * \hat{p}(y_k|x_j^k)$
 end for
 Normalize vector $(w_1^k, \ldots, w_N^k)$
end while

4.2 Implementation

In this section the implementation details of Algorithm 1 are discussed. y_k, the sensor readings, is the bit vector reported by sensors at time k. The object's movement f is approximated by taking x_j^k (the new particle) inside the area given by the following constraints:

- x_j^k has to lie outside the "minus" and "plus" convex hulls (from Theorem 2)
- x_j^k has to lie *inside* the circle of center S_+ and of radius the distance from S_+ to x_j^{k-1} (from Lemma 4), where S_+ can be any "plus" sensor at sampling times $k-1$ and k
- x_j^k has to lie *outside* the circle of center S_- and of radius the distance from S to x_j^{k-1} (from Lemma 4), where S_- can be any "minus" sensor at sampling times $k-1$ and k

The probability of the movement from x_j^{k-1} to x_j^k is approximated by

$$\hat{p}(y_k|x_j^k) = p_{slope}(x_j^k, y_k) \cdot p_{position}(x_j^k, y_k)$$

where p_{slope} is the ratio of possible slopes for the new position x_j^k and $p_{position}$ is a number that quantifies the relative location of the sensors, the old (x_j^{k-1}) and new (x_j^k) positions. More formally,

$$p_{position} = c \cdot \prod_{i=1}^{NS} \rho(S_i, x_j^{k-1}, x_j^k)$$

where c is a normalization constant, NS is the number of sensors and

$$\rho(S_i, x_j^{k-1}, x_j^k) = \begin{cases} 1, & \text{if } s_i^{(k-1)} \neq s_i^{(k)} \\ 1, & \text{if } s_i^{(k-1)} = s_i^{(k)} \text{ and } S_i, x_j^{k-1} \text{and } x_j^k \text{ respect inequalities (1)} \\ \frac{d(S_i, x_j^k)}{d(S_i, x_j^{k-1})}, & \text{if } s_i^{(k-1)} = s_i^{(k)} = 1 \text{ and } accept\_threshold < \frac{d(S_i, x_j^k)}{d(S_i, x_j^{k-1})} \leq 1 \\ \frac{d(S_i, x_j^{k-1})}{d(S_i, x_j^k)}, & \text{if } s_i^{(k-1)} = s_i^{(k)} = -1 \text{ and } accept\_threshold < \frac{d(S_i, x_j^{k-1})}{d(S_i, x_j^k)} \leq 1 \end{cases}$$

The acceptance criterion for x_j^k in Algorithm 1 is $p_{position} > accept\_threshold$. A small $accept\_threshold$ increases the estimation error, whereas a large $accept\_threshold$ (i.e. close to 1), increases the number of tries for finding a new particle (and thus the running time). A typical value for $accept\_threshold$ in our simulation is .8.

4.3 Experiments

To evaluate our approach, we implemented Algorithm 1 in **MATLAB** and performed extensive simulation on our implementation. All trajectories are taken inside the $[0, 1] \times [0, 1]$ square and thus the error measurements are relative to this square. Several types of trajectories have been considered: linear trajectories. trajectories with random turns and trajectories with "mild" turns (at each sensor readings the direction of the tracked object can vary from the previous one with at most $\pi/6$. All trajectories are piecewise linear and the distance traveled by the object between sensor readings is almost constant. A typical running example for a linear trajectory (figured by triangles) can be seen in Fig. 5. The distance traveled between sensor readings is $N(0.12, 0.02)$.

The simulation results (an example is given in Fig.5) suggested that this model can only give reliable information about the directionality. Indeed, the following indistinguishability result for position was obtained:

Proposition 5 *Let X_1 and X_2 two linear trajectories satisfying the following constraints:*

$$X_1'(t) = X_2'(t) \quad \forall t \Leftrightarrow \exists A \in \mathbf{R}^2 \text{ such that } X_2(t) = X_1(t) + A \quad A \cdot X_1'(t) = 0 \quad \forall t$$

i.e.,X_1 and X_2 are linear parallel trajectories like in Fig.6.

Then, no binary sensor can distinguish between trajectories X_1 and X_2. That is,

$$(S - X_1(t)) \cdot X_1'(t) = (S - X_2(t)) \cdot X_2'(t) \qquad \forall t$$

Proof.

$$\begin{aligned} (S - X_2(t)) \cdot X_2'(t) &= (S - X_1(t) - A) \cdot X_1'(t) = (S - X_1(t)) \cdot X_1'(t) - A \cdot X_1'(t) \\ &= (S - X_1(t)) \cdot X_1'(t) \end{aligned}$$

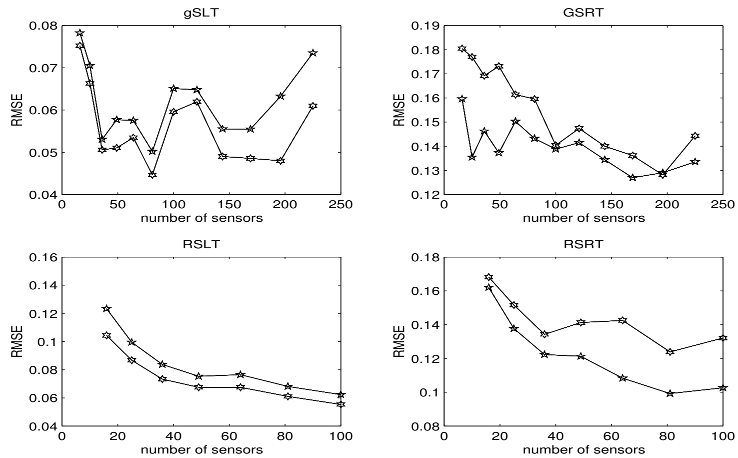

Figure 4: RMSE for random/grid sensorsXrandom/linear trajectory

□

One can also show that if no sensor placement can distinguish between two trajectories X_1 and X_2 then they must respect the conditions above. Indistinguishability for some sensor S is an equivalence relation but it seems hard to characterize the equivalence classes. For example two trajectories going clockwise(or anticlockwise) on two circles with the same center S, at the same angular velocity can not be distinguished by S.

5 Tracking with Range Bounds

5.1 Model

As Proposition 5 shows, no binary sensor can distinguish between parallel trajectories having the same velocities. In order our model to be consistent, an extra feature is necessary for the sensors. If the sensors are also able to signal the object's presence in their proximity, the indistinguishability problem is usually solved because in general, for random sensor positions, there are no two trajectories yielding the same sensor readings and entering the range of any sensor at the same time. We will assume that the sensors' locations and ranges are such that the ranges do not overlap. So, if a sensor "sees" the object no other sensor can see it at the same time.

5.2 Algorithm and Implementation

The Algorithm 2 uses Algorithm 1 as basis. If at some round the object is seen, for every particle which is not inside the range it shifts ancestors of this particle as far as the last time the object was spotted by proportional amounts.

5.3 Experiments

If we assume the sensors have the ability to report the presence of the object in their proximity, then a good measure of the performance of the algorithms the relative error *after* the object is first spotted. But there is another question which has to be answered first: "After how many turns does the object is first spotted given a sensor layout?". Some simulation results

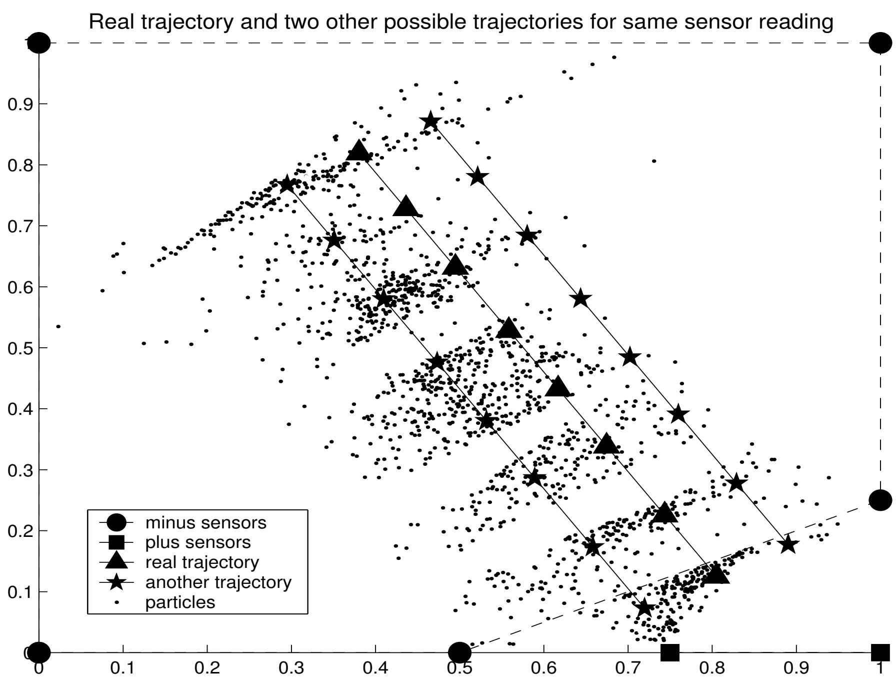

Figure 5: Running example for sensors with no range bounds

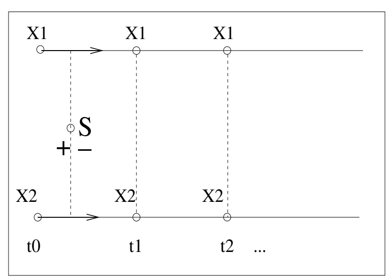

Figure 6: Indistinguishability between linear parallel trajectories

are shown in Fig. 7. Fig. 7 shows how many trajectories out of 100000 have entered a sensor range after k turns, where k goes from 1 to 800. The trajectories were generated as follows: the distance traveled between sensor readings is $N(0.02, 0.001)$ and the changes in direction are "mild" (that is, the direction can change with at most $\pi/6$ at each sensor reading. The results are for 25 and respectively 100 sensors. At right, a small value for the range is taken (the ranges cover less than 10% of the whole area) and at left, a big value for the range is considered (the ranges cover about 70% of the whole area). The graphs suggest that the distribution of the number of turns that pass until an object is first spotted is exponential. *Add some more here ...*

A running example for Algorithm 2 can be seen in Figure 8.

Algorithm 2 Algorithm for Binary Sensors with Range

- Use Algorithm 1 as basis.

if sensor S sees the object **then**
    for all accepted particles P not inside the range of S **do**
        Let P' (a new particle) be the intersection between the range of S and semi-line $(PS]$
        Let $P_1, \ldots, P_k$ be the ancestors of P since the last time the object was spotted.
        for $i = 1$ to k **do**
            $P_i = P_i - (P - P')/(k+1)$
        end for
    end for
end if

6 Conclusions and future work

References

[1]

[2] D. Crisan and A. Doucet. A survey of convergence results on particle filtering for practitioners, 2002.

[3] W.E.L. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site.

[4] P. Clifford J. Carpenter and P. Fearnhead. An improved particle filter for non-linear problems. In *IEE proceedings - Radar, Sonar and Navigation*, pages I46:2–7, 1999.

[5] D. Salmond N. Gordon and A. Smith.

[6] Lynne E. Parker. Cooperative motion control for multi-target observation. In *Proc. of IEEE International Conf. on Intelligent Robots and Systems*.

[7] Michael K. Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446), 1999.

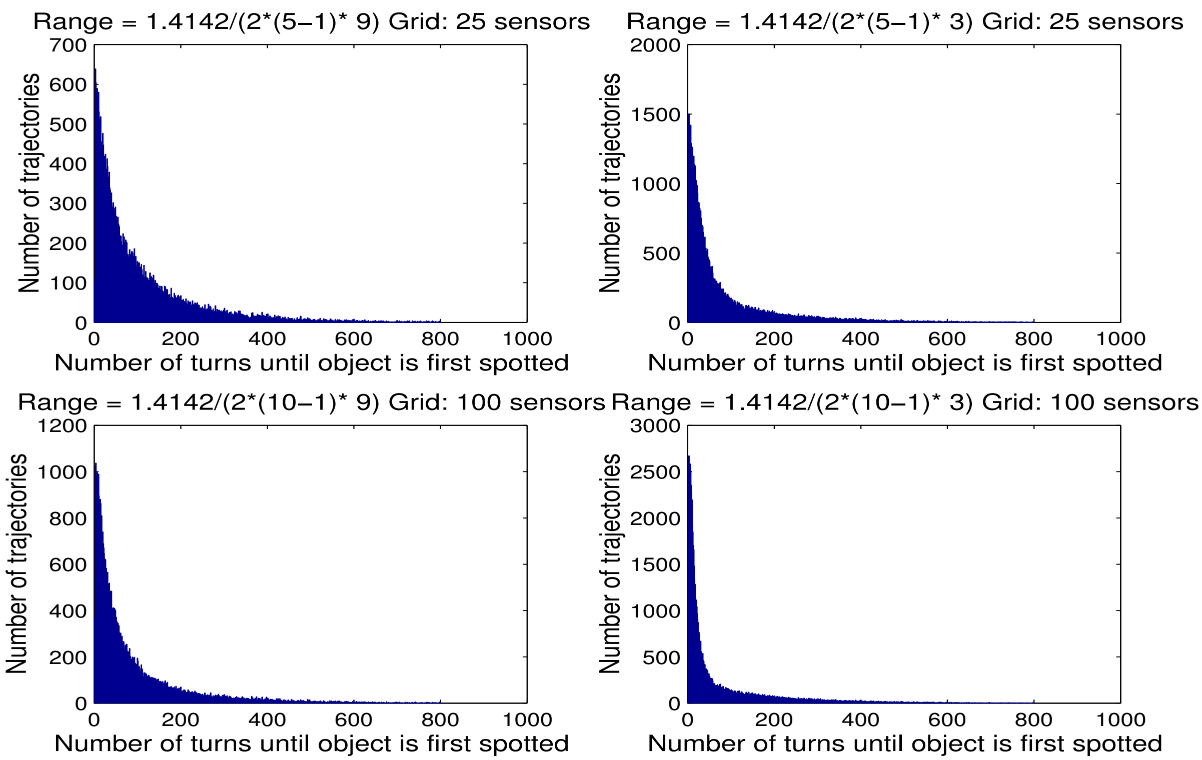

Figure 7: Number of turns until the object gets first time in a sensor range

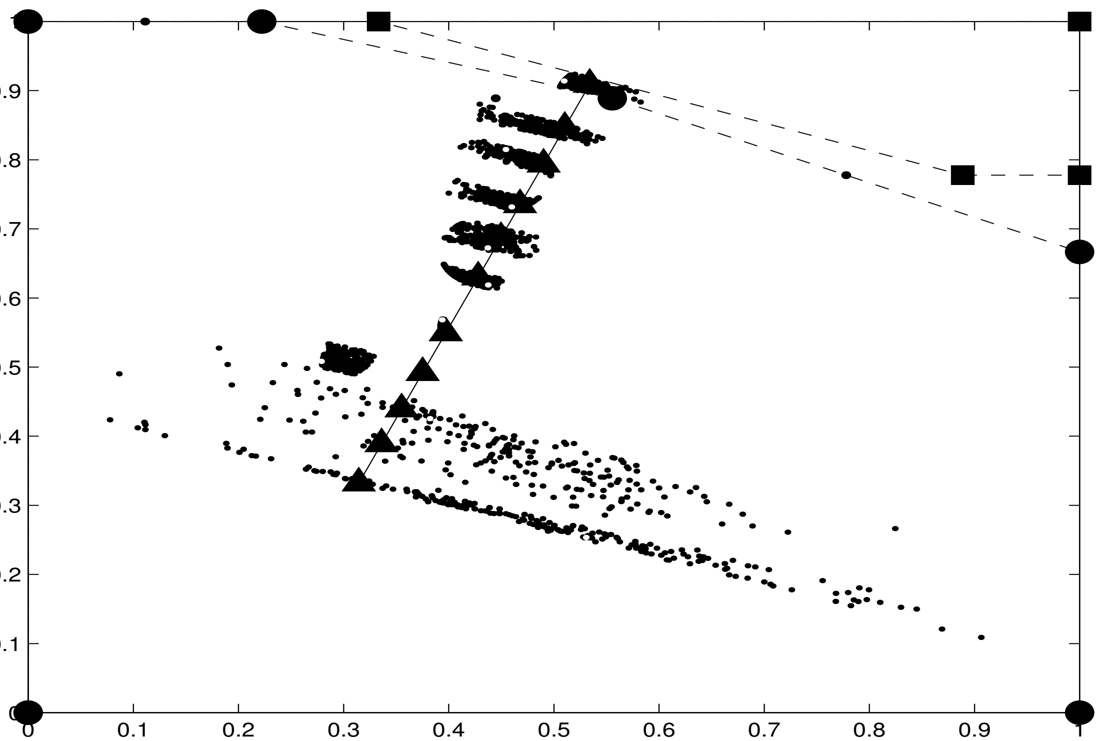

Figure 8: Running example for Algorithm 2