UAV based search for a radio tagged animal using particle filters

André Posch
Institute of Flight Mechanics and Control *
Universität Stuttgart
70569 Stuttgart, Germany
ifraposc@ifr.uni-stuttgart.de

Salah Sukkarieh
Australian Centre for Field Robotics
The University of Sydney
Sydney, NSW 2006, Australia
salah@acfr.usyd.edu.au

Abstract

Radio animal tracking is usually carried out by surveying with an antenna and receiver. The researcher can determine the position of the animal by rotating the directional antenna and listening to the signal strength while walking around. Radio tags for animals have to be quite small with low weight and therefore have limited signal power resulting in a short range. This procedure could be tedious and therefore we present an alternative method using unmanned aerial vehicles with similar sensor equipment. The search algorithm uses a particle filter for representation of the probability density function of the animal's states. For the guidance of the UAV two algorithms are compared: One is based on a dynamic grid overlay for the particles to quickly find the best destination while the other one is a more optimal but slower algorithm based on maximising the information gain. As example we show simulation results of the search procedure. Finally we present results from software-in-the-loop and hardware-in-the-loop tests using the cloud cap autopilot.

1 INTRODUCTION

One of the interesting applications is to use radio tracking to study the habitat of a certain wild species. The manual procedure is as follows: Using a directional antenna (the Yagi design is the most common) and a receiver with amplifier, one tries to find the direction which gives the strongest signal. More bearing measurements can be taken by moving to other positions perpendicular to the line of the target. It is then possible to calculate the target position by triangulation. However this only works when the signal is receivable. One must also take the weight constraint of the radio tags into consideration. In reality the tags must be very light so they do not interfere with the animal. Therefore they have only a small battery and emit a signal with very limited power resulting in short range. Another problem is that the radio source and the receiver are both next to the ground, which results in even shorter range and more multipath errors. Utilizing Unmanned Aerial Vehicles (UAVs) for this task gives some advantages. (i) The speed of the UAVs is many times higher than that of a human. (ii) UAVs are not affected by rough terrain. (iii) The signal range from ground to air is much better, in most cases many times larger than the range of a ground to ground transmission.

1.1 Contribution

The low cost single antenna setup used in this paper gives neither absolute bearing nor range information in contrast to more expansive and complicated direction finding antennas [Briner *et al.*, 2003]. There are many locations matching a certain signal strength. To use this highly nonlinear and non-Gaussian information a particle filter is utilized. A model for the sensor is developed and a fast and simple guidance algorithm is designed to calculate the best destination for the UAV in realtime.
The paper is organized as follows. After describing particle filters and their advantages over traditional filter methods the single steps in the particle filter are described. Next the guidance algorithms are presented, which either utilizes a grid based on the information in the particle set or compares the information gain from different plans. The next section describes the motion and sensor models used in the implementation and gives an overview over the whole algorithm. Finally simulation results are presented and a brief conclusion and ideas about future work are given.

2 PARTICLE FILTER

Bayesian methods are the common choice for state estimation problems with uncertainty. They are used to

*work was carried out during visit at the Australian Centre for Field Robotics

calculate the probability density function (PDF) of the system states by using all known information. For the linear and Gaussian case optimal solutions are known (the Kalman filter). For nonlinear problems the extended Kalman filter can be used, which linearizes the problem about the predicted state to make application of the Kalman filter possible. But if the PDF is highly non-Gaussian (as presented in this paper) the Kalman filter performs poorly. An approach are grid based filters, but they tend to be accompanied by a high computational effort. A cleaner solution for nonlinear non-Gaussian problems are particle filters (also called sequential Monte Carlo methods) which represent the PDF as a set of (weighted) samples. As the number of samples becomes large they provide a good representation of the PDF. A good comparison is given in [Arulampalam *et al.*, 2002] and [Thrun *et al.*, 2005].

2.1 Recursive Bayesian estimation

To construct the posterior PDF one uses all available information including all observations made up to that time. An optimal solution for the state could be calculated from that PDF. Because of memory and computation power limits it might be impossible to keep all information for every point in history and calculate the PDF from this information every time an observation is made. Therefore it is much better to use a recursive approach and process only the new information that becomes available. A filter using this concept consists of at least a prediction step and an update step. The prediction step uses the system model (including noise) to forecast (and spread) the PDF. The update step then modifies (and shrinks) the PDF by integrating the new information from the observation using Bayes theorem.

2.2 Particle Filtering

Prediction

To include the knowledge about the system we predict [Marchetti *et al.*, 2007] the PDF

$$p(x_k|z_{0:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{0:k-1})dx_{k-1} \quad (1)$$

using the system motion model (in our case the target motion model) given in discrete form as

$$x_k = f(x_{k-1}, u_k, s_k, \Delta t) \quad (2)$$

where x_k are the system states at time k, u_k are the model inputs, s_k is the uncertainty in the model, Δt is the length of the time interval and $z_{0:k}$ are the observations made up to time k. Application to the particles is simple; the motion model on each of the particles.

Roughening

If the system motion model has little or no noise it appears (as further explained later in the resampling step) that the filter will never lead to the correct solution regardless of how good the observations are. To overcome this, one option is the introduction of a roughening step, which applies an artificial noise to the particles. This noise could be Gaussian with a constant or dynamic covariance matrix (which is a tweaking parameter). The roughening used in the simulation has a dynamic Gaussian noise based on the standard deviation of the whole particle set.

Update

Using the predicted PDF and Bayes' rule

$$p(x_k|z_{0:k}) = \frac{p(z_k|x_k)p(x_k|z_{0:k-1})}{p(z_k|z_{0:k-1})} \quad (3)$$

with the normalizing constant

$$p(z_k|z_{0:k-1}) = \int p(z_k|x_k)p(x_k|z_{0:k-1})dx_k \quad (4)$$

we can update [Marchetti *et al.*, 2007] the prior PDF using the likelihood function of the observation $p(z_k|x_k)$. For the particle filter this is approximated by

$$p(x_k|z_{0:k}) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \quad (5)$$

with the weights

$$w_k^i \propto w_{k-1}^i p(z_k|x_k^i) \quad (6)$$

if we choose the importance density to be the prior [Ristic *et al.*, 2004].

Normalize

To represent a valid PDF the accumulated weight of all particles must be equal to 1. As the update step changes the total weight, a normalization of the weights must be performed:

$$w_k^i = \frac{w_k^i}{\sum_{j=1}^{N} w_k^j} \quad (7)$$

Resampling

After a number of filter steps it appears that most particles have a very low weight. This is known as the degeneracy problem [Doucet *et al.*, 2001]. To measure the level of degeneracy, one can use the number of effective particles [Ristic *et al.*, 2004], given as

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} ({}^i\tilde{w}_k)^2} \quad (8)$$

If N_{eff} drops below a certain threshold (e.g. half of all particles) it is necessary to perform a resampling

step. There are several algorithms known to perform the resampling. We use systematic resampling from the package pfLib [Chen *et al.*, 2007]. The algorithm redistributes the particles by removing particles with very low weight and adding particles at positions of particles with high weight. This resembles natural selection and results in having more particles in areas were they are needed. The problem with this is obvious: Without some random noise in the prediction step this results in having all particles concentrated on a few spots. This means if there was no particle next to the real target position in the initial particle distribution, the update step will fail eventually. Therefore it is necessary to introduce the roughening step to spread the particles.

Random Injection

Furthermore under some circumstances (e.g. after some false positive detections) it could happen that particles in a wide area around the real target position are removed from the particle set. Under bad conditions the roughening step is not enough to recover from that problem. To deal with this problem a random injection step [Marchetti *et al.*, 2006] [Thrun *et al.*, 2005] is introduced which injects some particles with random states. This raises two questions: How many particles should be injected at each step and from which distribution should they be drawn? Instead of adding a fixed amount of particles it's better to make it dependent on the behaviour of the filter. A better approach is to monitor the weights after the update step. If there are no particles next to the real state the weights will be very low. To remove effects of a large sensor noise the average weight over a few steps is compared with the average weight over a long term. The probability for random injection is then calculated from this measure.

$$\begin{aligned} w_{avg,k} &= \frac{1}{N}\sum_{j=1}^{N} w_k^j \\ w_{slow,k} &= w_{slow,k-1} + \alpha_{slow}(w_{avg,k} - w_{slow,k-1}) \\ w_{fast,k} &= w_{fast,k-1} + \alpha_{fast}(w_{avg,k} - w_{fast,k-1}) \\ P_{injection,k} &= \max\left(0, 1 - \frac{w_{fast,k}}{w_{slow,k}}\right) \end{aligned} \quad (9)$$

A good distribution to use would be one based on the current observation. But this is not practical in our case. We use the particle mean and standard deviation to generate a random state for the injected particle.

3 GUIDANCE ALGORITHM A - GRID BASED

For performance reasons a simple but effective guidance algorithm is implemented. The first step is to build a grid from the particle set which in principle holds the total number and weight of the particles in each grid cell. Next the cells are modified using the distance to the UAV. This gives the best destination for the UAV and is the base for calculating the inputs. As this method is not very computation intensive it can be evaluated with a high frequency.

3.1 Grid Approximation

Each grid cell holds the accumulated weight of all particles inside that grid cells area, calculated by

$$w_{ix,iy}^{cell} = \sum_{i|x^i \in cell} w^i \quad (10)$$

with $w_{ix,iy}^{cell}$ be the weight of grid cell ix, iy. Furthermore the center of mass of the contained particles is stored to get a better representation of the particle distribution. The number of the grid cells is a tuning parameter. The size (calculated from the number) should be smaller than the area of detection of the sensor. If not then the best destination may be in the center of a grid cell and therefore the UAV will be stuck in an infinite loop and never find the target.

3.2 Picking Best Destination

The simplest approach would be to head to the grid cell with the highest accumulated particle weight. It's obvious that this is far from optimal as the time to reach the destination must be taken into account. Therefore two modifications are made. First each cells accumulated particle weight is increased by the weights of it's surrounding cells times the distance to these cells times a fixed factor γ. α is a tuning parameter which limits the considered cells.

$$w_{ix,iy}^{cell} = w_{ix,iy}^{cell} + \gamma \sum_{\substack{jx\in[ix-\alpha..ix+\alpha] \\ jy\in[iy-\alpha..iy+\alpha]}} w_{jx,jy}^{cell} \left| x_{ix,iy}^{cell} - x_{jx,jy}^{cell} \right| \quad (11)$$

This gives more priority to cells which are in an area with lots of particles around. The second step is to modify each cells weight using the distance to the UAV times a proportional factor κ

$$w_{ix,iy}^{cell} = \kappa \left| x_{ix,iy}^{cell} - x^i \right| w_{ix,iy}^{cell} \quad (12)$$

where $x_{ix,iy}^{cell}$ is the center of mass of grid cell ix, iy.

3.3 Calculating Inputs

For the simulation it's necessary to calculate the inputs for the UAV motion model. This is carried out in equation 13, which just tries to bring the line of sight ϕ_k^{LoS} to zero.

$$\begin{aligned} \phi_k^{LoS} &= pp\left(\gamma_k^u - atan2\left(-y_k^u + y_k^{dest}, -x_k^u + x_k^{dest}\right)\right) \\ \alpha_k^u &= (-1)\cdot sgn\left(\phi_k^{LoS}\right)\cdot min\left(\frac{\left|\phi_k^{LoS}\right|}{\Delta t}, \alpha_{max}^u\right) \end{aligned} \quad (13)$$

Where α^u_{max} is the maximum turn rate of the UAV, γ^u_k is the UAV heading angle, $[x^{dest}, y^{dest}]$ is the center of mass of the grid cell selected as destination and pp is a function converting the range $[0..2\pi]$ to $[-\pi..\pi]$. In the real life application an auto pilot is used, which only requires waypoints. The low level control is then carried out by the autopilot.

4 GUIDANCE ALGORITHM B - INFORMATION GAIN

The second algorithm developed tries to maximise the gain of new information for future steps. The planing step runs every few filter steps and calculates the change in total particle weight for different plans. Assuming that the target is not detected it predicts the sensor observations for the next steps for a set of predefined plans. It is obvious that this requires a lot of computation time, which increases exponentially with the length of the planing horizon and number of plans evaluated. To accelerate the calculation it is possible to use only a subset of the whole particle set - with the drawback of being less optimal.

4.1 Plans

The selection of plans is controlled by three parameters: First is the number of choices $n_{choices}$ for the UAV inputs [full turn left, go straight, full turn right]. Second is the number of decisions $n_{decisions}$, which means how many changes to the input are considered. Last is the number of steps n_{steps} one decision is kept. The total number of filter steps to evaluate in the planing algorithm is therefore:

$$n_{steps} \cdot n_{choices}^{n_{decisions}} \tag{14}$$

4.2 Optimal Plan Selection

Based on the current state of the particle set the behaviour of the particles for each of the plans is computed. As we do not know where the target is, it is assumed to be outside the planing horizon. Furthermore some steps of the filter are not necessary or improper for planing. Normalization and resampling does not take place as it would make comparision of performance of the different plans impossible. Also it's not necessary to maintain mean and standard deviation of the particle set, roughen the particles or perform random injection as this will be performed different later. After predicting the behaviour of every single plan the total change in particle weight is compared and the plan with the greatest change is picked as the optimal plan and saved.

4.3 Calculating Inputs

Calulating the inputs for the UAV is straightforward: since the optimal plan is saved we pick the appropriate command for the current time while this plan is valid.

5 IMPLEMENTATION

5.1 Motion Models

UAV Motion Model

The UAV motion [Furukawa *et al.*, 2007] is represented by

$$\begin{aligned} x^u_{k+1} &= x^u_k + \Delta t \cdot v^u_k \cos\gamma^u_k \\ y^u_{k+1} &= y^u_k + \Delta t \cdot v^u_k \sin\gamma^u_k \\ \gamma^u_{k+1} &= \gamma^u_k + \Delta t \cdot \alpha^u_k \end{aligned} \tag{15}$$

with the UAV velocity v^t_u, UAV turn rate α^t_u and time step Δt. For the UAV used in the simulation the parameters are

$$\begin{aligned} v^u_k &= 25.7m/s \forall k \\ \alpha^u_{max} &= 0.1rad/s \end{aligned}$$

Target Motion Model

The target motion is represented by

$$\begin{aligned} x^t_{k+1} &= x^t_k + \Delta t \cdot v^t_{x,k} \\ y^t_{k+1} &= y^t_k + \Delta t \cdot v^t_{y,k} \\ v^t_{x,k+1} &= v^t_{x,k+1} + \Delta t \cdot a^t_{x,k} \\ v^t_{y,k+1} &= v^t_{y,k+1} + \Delta t \cdot a^t_{y,k} \end{aligned} \tag{16}$$

with the target velocities v^t, target accelerations a^t and time step Δt. For the first part of the simulation the target is assumed to stay at it's position and therefore $v^t_k = 0$ for all times k. For the second part the target is accelerating randomly using $a^t_k = \mathcal{N}(0, \sigma_{targetAccl})$

5.2 Observation

The animal will be fitted with a radio tag commonly used in radio animal tracking. This emits a low power VHF radio signal. The radio wave is propagated through the air, losing power with distance. A simple Yagi antenna on the UAV will receive the signal which is then recognized by the receiver (biotrack Sika [Biotrack, 2009]). The receivers audio output is then sampled by a microcontroller and transfered to the PC104 via RS232. The PC104 also sets receiver gain and desired frequency via RS232.

Radio Emitter

We use a Pip two-stage transmitter manufactured by biotrack and weighing only $4.5g$. It can emit $0.5mW$ on the desired frequency of $173Mhz$. Fitted with an Ag357 battery it can transmit up to 5 months. For simplification we assume the emitting antenna to be perfectly omni-directional.

Wave Propagation

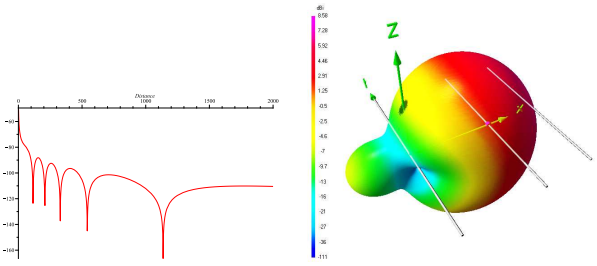
We use a 2-wave model (free space and reflected wave) [Struzak, 2004] [Circuit Design, 2009] for the wave propagation, which incorporates among others RX/TX antenna height and wavelength. First the phase difference $\Delta\phi$ is calulated by

$$\begin{aligned} l &= \sqrt{\left(d^2 + (h_t + h_r)^2\right)} - \sqrt{\left(d^2 + (h_t - h_r)^2\right)} \\ \Delta\phi &= \frac{2 \cdot \pi \cdot l}{\lambda} \end{aligned} \tag{17}$$

using ground distance d, antenna heights h_t and h_r, path difference l and the wavelength λ. Then the output power P_{read} is calculated using

$$\begin{aligned} P_{rp} &= P_s \cdot L_{ct} \cdot L_{ot} \cdot G_t \\ E_0 &= \sqrt{\frac{30 \cdot P_t \cdot G_t}{d}} \\ E &= 2 \cdot E_0 \cdot \left|\sin\left(\frac{\Delta\phi}{2}\right)\right| \cdot L_e \\ P_r &= \frac{E^2 \cdot \lambda^2 \cdot G_r}{480 \cdot \pi^2} \\ P_{read} &= P_r \cdot L_{cr} \cdot L_{or} \end{aligned} \tag{18}$$

with the supply power P_s, effective isotropic radiated power P_{rp}, received power P_r, cable and other losses $L_{ct}, L_{ot}, L_{cr}, L_{or}, L_e$, antenna gains G_t and G_r and electric field intensities E_0 and E. An example with $P_t = 0.5mW$, $h_t = 5m$, $h_r = 200m$ is shown in Fig. 1(a).



(a) Wave Propagation (b) Yagi Antenna Gain

Figure 1: Signal strength models

Another option would be to use a wave propagation model which incorporates information about the terrain like the Longley-Rice model [A.G. Longley, 1968] or ITU-R P.1546 [ITU, 2001]. The major drawback is that these models are computational expensive and cannot be evaluated for each particle at every filter step. Because of the usage of a terrain model it is not even possible to precompute the results and store them in a lookup table.

Antenna

The antenna is a common $\frac{\lambda}{2}$ directional yagi design. It was modeled in the software 4nec2 to create a lookup table for the gain in all directions to speed up filter updating. With the model used, the gain is around $8dB$ and back-to-front ratio around $14dB$ with a $3dB$ beamwidth of 80°. A visualization of the alt/az gain is shown in Fig. 1(b). The directional yagi antenna design was choosen because it provides more diversity in signal strength when the plane changes orientation. An omni directional antenna will not give any more information on rotation. Simulations carried out with a moving target show the difference between real target position and estimated target position for an omni gain antenna vs. a directional (yagi) design. Apart from the antenna model all simulation parameters were identical. It can be seen in Fig. 2 which shows a box plot where the central mark is the median and the edges of the box are the 25th and 75th percentiles, that the error with the omni antenna is approximately twice as much as the error with the yagi antenna.

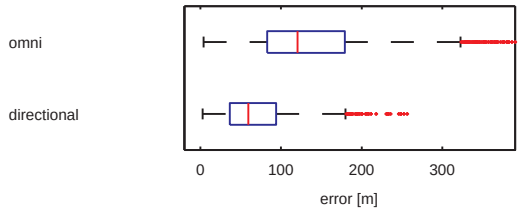


Figure 2: Comparing the tracking error in target position using Omni Gain vs. Directional Gain antenna

Observation Likelihood

The likelihood used in the update step $p(z_k|x_k)$ for each particle is different for the detect and no-detect case [Bourgault *et al.*, 2004]. For each particle the received power P_p is calculated using the wave propagation and antenna model. If the target is detected the likelihood is calculated by

$$1 - \Gamma_{nodet} + \Gamma_{det} \cdot \mathcal{N}\left(P_p - P_o, \sigma\right) \tag{19}$$

with Γ_{nodet} and Γ_{det} factors representing the influence of new observations. The standard deviation σ depends on many different factors like terrain, multipath effects, errors in the models etc. and is estimated during tests. P_o is the signal power observed by the sensor.
If the target is not detected the likelihood is calculated by

$$1 - \min\left(1, \max\left(0, \Gamma_{nodet} \cdot (P_p - P_t)\right)\right) \tag{20}$$

where P_t is the threshold for detection, determined by the minimal signal power recognized by the receiver. This function defines a linear gradient in the field-of-view of the sensor with values in the range $[0; 1]$.

Environmental radio noise

In the simulation the artificial radio noise of the environment [Hagn, 1980] is modeled as follows

$$\begin{aligned} \overline{n} &= -174 - 27.7 * \log_{10}(f_{Mhz}) + \eta_{env} \\ \sigma_n &= \frac{7.2 \cdot (250 - f_{Mhz}) + 3 \cdot (f_{Mhz} - 102)}{148} \\ n_k &= \mathcal{N}(\overline{n}, \sigma_n) \end{aligned} \quad (21)$$

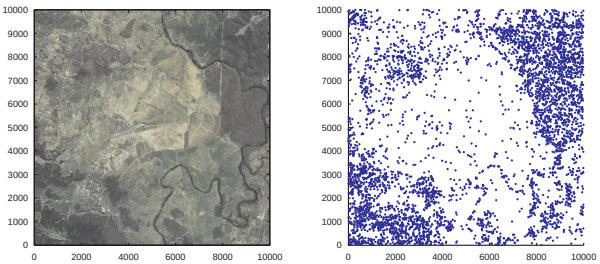
with noise mean n_{mean}, frequency $f_{Mhz} \in [102; 250]$ and environment constant $\eta_e nv$ which is 67.2 for rural areas. The result is converted to mW and added after calculating the propagated signal. Atmospheric and galactic noise are not relevant for the frequencies used.

5.3 Initial Particle Distribution

The initial particle distribution can be built from a uniform distribution, but it is more likely that some prior knowledge about the initial distribution exists. In our case the animal is known to stay in the woods or next to water most of the time. A satellite image of the search area is used to classify regions j of high or low probability. To build the initial particle distribution the desired number of particles d^j for each region is calculated. The particles are placed randomly on the map while comparing the actual number of particles n^j for each region with d^j. If $n^j > d^j$ the particle is rejected and placed again. One result can be seen in Fig. 3. This is shown in Algorithm 1 where N is the total number of particles in the filter.

Algorithm 1: Build Initial Particle Distribution

```
calculate desired particles d^j
for i=1 to N do
    repeat
        draw x_0^i from uniform distribution
        get region j from x_0^i
        increase n^j
    until n^j ≤ d^j ;
end
```



(a) Satellite Image (b) Particle Distribution

Figure 3: Initial Particle Distribution

5.4 Algorithm

In summary the complete algorithm for searching a target with an UAV is shown in Algorithm 2. This includes simulation of the sensor data.

Algorithm 2: Complete Search

```
initialize particles from initial distribution
while true do
    perform guidance
    evaluate UAV motion model
    evaluate Target motion model
    simulate UAV sensor observation
    predict particles
    roughen particles
    foreach particle i with weight w^i do
        if Target detected then
            calculate likelihood l^i from threshold
        else
            calculate likelihood l^i from observation
        end
        w^i = w^i l^i
    end
    update w_{fast,k}, w_{slow,k} and P_injection
    normalize particles
    perform systematic resampling
    calculate mean and standard deviation
    inject random particles based on P_injection
end
```

Algorithm 3 shows the Grid Based guidance law.

Algorithm 3: Guidance - Grid Based

```
calculate base grid
smooth grid
weight grid cells by distance
set UAV destination
update UAV inputs
```

Algorithm 4 shows the Information Gain guidance law.

6 SIMULATION RESULTS

All simulations were carried out using Algorithm 2. The initial particles were drawn, as described above, from a satellite image showing the ACFR's flight test facilities at Marulan, NSW or from a uniform distribution. The size of the area used for the simulations is $10 \times 10km$. The target antenna is assumed to be located $1m$ above the ground and omni directional with no gain. The detection threshold for the receiver is $-130dBm$ and σ is assumed to be $5dB$. Furthermore the accumulated loss (cable, other, environment) is set to $12dB$. The UAV always takes of from the runway at $[6444, 2097]$, heading south east.

Algorithm 4: Guidance - Information Gain

```
if t_PlanStart + planinghorizon ≤ t then
    get UAV inputs from last plan
else
    foreach possible plan i do
        for i_dec = 1 to n_decisions do
            for i_steps = 1 to n_steps do
                set UAV inputs from plan i
                evaluate UAV motion model
                predict particles
                update weights
            end
        end
        caluclate total weight
    end
    save plan with lowest total weight
    set t_PlanStart = t
end
```

6.1 Performance

For each case 100 flights were simulated to compare performance of the different guidance algorithms and initial distributions. Each simulation was stopped when a standard deviation of $50m$ was reached. For these tests 1000 particles where used.

Initial Distribution from Map, Target Position as expected

In this case the initial particles were drawn from a map and the target is located in an area with high initial probability (at $[8000, 8000]$). Although the information gain planer should be optimal it performs worse than the grid based guidance (as seen in Fig. 4). The reason is the very limited planing horizon, which leads to an only locally optimal selection of the best path. On the contrary the grid based guidance is locally far from optimal, but uses the global knowledge much better.

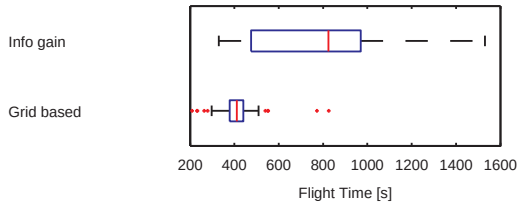


Figure 4: Mean Flight Time (Map, as expected)

Initial Distribution from Map, Target Position not as expected

In this case the initial particles were drawn from a map and Target is located in an area with low initial probability (at $[2000, 8000]$). It can be seen in Fig. 5 that in this case both planers behave quite similiar. On further inspection the reason is that the real target position lies out of a pure case on the way the information gain planer chooses. The grid based planer flies at first to the areas with high probability, which takes some time. It is pure coincidence the information gain planer performs so well in this case.

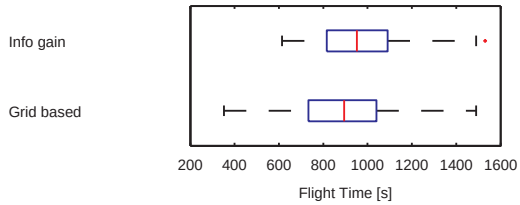


Figure 5: Mean Flight Time (Map, not as expected)

Initial Distribution Uniform

In this case the initial particles were drawn from a uniform distribution and Target is located at $[2000, 8000]$. One might except the planers behave quite similiar in this case, but the grid based planer is again working better, as seen in Fig. 6. The reason for this is not obvious. Maybe the limit control choices together with the keeping time for the information based planer cause unnecessary detours. Another reason might be the typical behaviour caused by the limited planing horizon, where the planer chooses a path which leads to an area with already low probabilities.

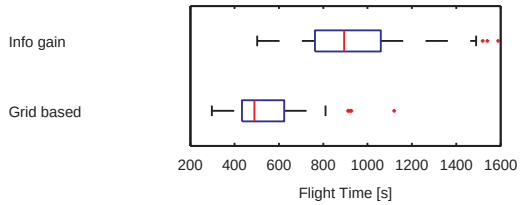


Figure 6: Mean Flight Time (Uniform)

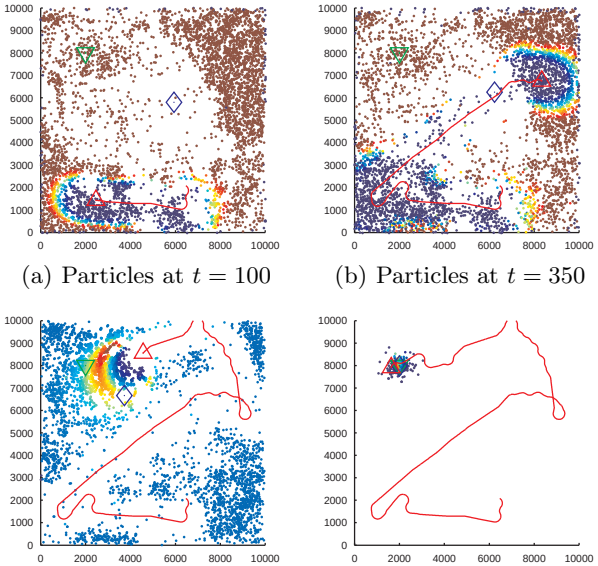
6.2 Examples

In the images the green triangle is the target and the green path is the target's path if applicable. The red triangle is the UAV with shown orientation and it's flight path marked in red. In case of the information gain planer the yellow line is the planed path. The blue rhomb or blue path is the particle mean. Particles are shown as dots and the particle weights get lower towards blue. For these simulations 5000 particles where used to get a better visualisation.

Fixed Target, Grid Based Guidance

The first example shows a search using the grid based guidance. The target is located at $[2000, 8000]$ (which is

an area with low probability) and not moving. Δt is $0.5s$ and the grid spacing is $1km$ giving 10×10 cells. Fig. 7 shows the search. After take off the UAV heads to the south west area as it has a high probability and is closer than the north east area. After some observations most particles have low weight and the UAV heads north east. After a few observations and several resampling steps a lot of particles are located in the north west and the UAV finally finds the target there. It can be seen that the grid based guidance algorithm makes use of the global particle distribution. Computation time was $10.4ms$ per filter step on a 2.8 Ghz Pentium 4 with 1 GB of RAM running Windows XP which is good enough for implementing it on the UAV's PC104 for real time use. Setting the number of particles to 1000 has no great effect on the filter. It performs similar by reducing the computation time to $4.53ms$.



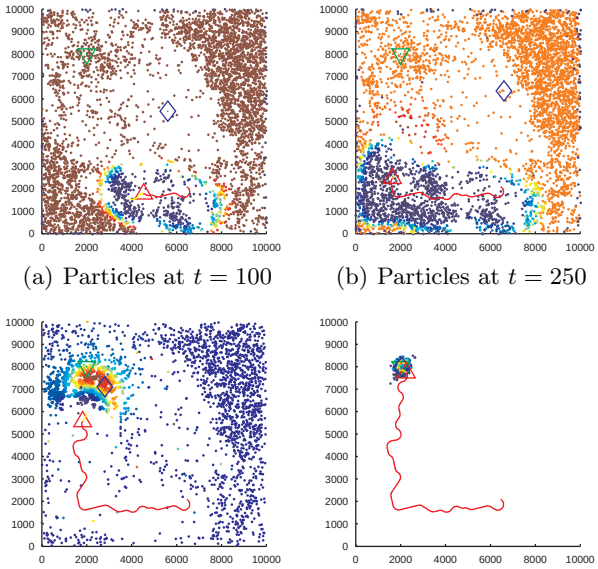
(a) Particles at $t = 100$ (b) Particles at $t = 350$

(c) Particles at $t = 550$ (d) Particles at $t = 625$

Figure 7: Fixed Target, Grid Based Guidance

Fixed Target, Information Gain Planer

The second example shows a search using the information gain planer. The target is again located at $[2000, 8000]$. There are 3 choices for the planer and 4 decisions which are kept for 16 filter steps. This gives a total of 81 plans to compare (or 1296 filter steps to calculate) at each planing step. The planing horizon is 32 seconds. Fig. 8 shows the search. After take off the UAV heads to the south west area as it gains most information on this way. But instead of heading to the north east afterwards it heads north, following the path of greatest information gain and finding the target there. Again it's pure coincidence that the target is found soon. It can be seen that the information gain planer is only local optimal because of its limited planing horizon. Computation time was $42.7ms$ per filter step. Setting the number of particles to 1000 has great impact on the filter as the number of particles used for the planing step is 10% of the total particles, in that case this leads to only 100 particles used for planing, which isn't a good base anymore. Apart from that the computation time would be reduced to $12ms$.



(a) Particles at $t = 100$ (b) Particles at $t = 250$

(c) Particles at $t = 400$ (d) Particles at $t = 500$

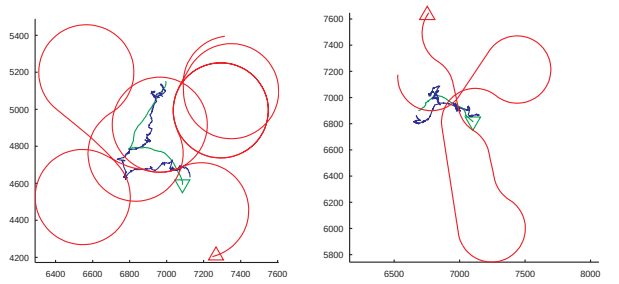
Figure 8: Fixed Target, Information Gain Planer

Moving Target, Information Gain vs. Grid Based

The choice of the planer has no impact on the tracking problem. As the target is found it makes no difference which path is choosen as long as the UAV moves around the target. Both planers choose similiar paths. While the grid based planer circles around the center of the particle set, the information gain planer moves a little bit further away. This happens because the planing always assumes no detection. The behaviour of the information based planer might be improved by incorporating the information if the target is already detected. But this means switching between a searching and a tracking phase, which does not fit in the concept of a unified approach. Fig. 9 shows the two planers tracking a moving target.

Moving Target, Omni vs. Directional Antenna

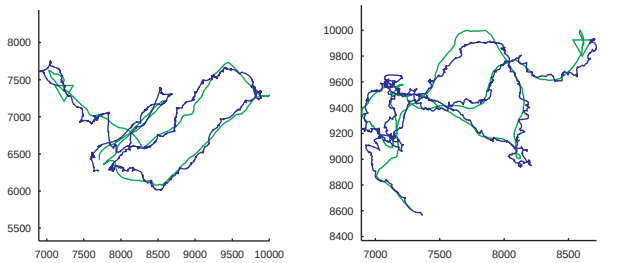
This example shows the impact of the antenna design on the mean error while tracking. As discussed in the antenna section above, the error is larger for an omnidirectional antenna. It can be seen in Fig. 10 that the



(a) Grid Based Guidance (b) Information Gain Planer

Figure 9: Moving Target, Information Gain vs. Grid Based

target mean better represents the real target position when using a yagi antenna. For this simulation the grid based guidance was used and the path is only shown for the tracking phase.



(a) Omni Antenna (b) Directional Antenna

Figure 10: Moving Target, Omni vs. Directional Antenna

7 SOFTWARE / HARDWARE IN THE LOOP TESTS

7.1 SIL Setup

The hardware setup for the software-in-the-loop simulations consists of one PC running the aircraft dynamics, ground station and piccolo autopilot simulators, one PC running the simulated onboard pc-104 under QNX [Krten, 2008] and one PC running the windows version of the Momentics Ide, all connected via gigabit ethernet. The simulated aircraft dynamics closely represent the dynamics of the plane used in the real flight experiments.

7.2 HIL Setup

In the hardware-in-the-loop simulations the ground station and piccolo autopilot simulators are replaced by the real autopilot hardware. Communications to the groundstation take place over a radio link. The onboard computer communicates with the piccolo autopilot via RS232. As a PC104 was not available at this time, the PC from the SIL simulations is used again as an onboard computer. Sensor and aircraft dynamics are still simulated.

7.3 Modifications

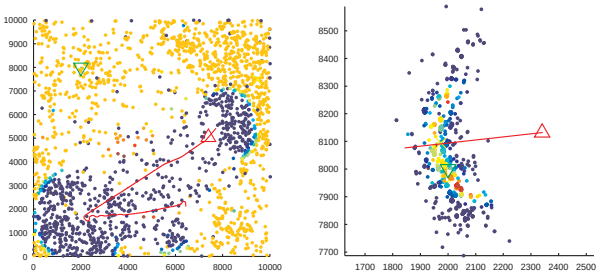
The software used in the simulation stage had to be slightly modified to run on QNX. Because the tests were carried out for the grid based planer only, the simulated UAV model was not required anymore, as position and orientation information is directly acquired from the piccolo autopilot. Instead of using only position and bearing of the UAV (as in the dubin's car model used in the simulations before), now a full 6-DOF information is used to calculate antenna elevation and azimuth with respect to the target. The aircraft dynamics were neglected in the simulations done before, but now have to be taken into account as the piccolo heading controller has a limited bandwidth. Therefore the rate of sending heading commands to the autopilot is now limited. The filter is not running at a fixed frequency anymore, but as fast as possible. Time results for the SIL simulations are a Δt with mean $17ms$ and standard deviation of $7ms$ on the a 3Ghz x86 PC. Main reasons for the variation are the simulated autopilot (which is not running on a realtime OS), the resampling step and the random injection step, which only take place when a certain threshold is reached. Time results for the HIL simulations were a mean of $43ms$ and standard deviation of $22ms$. The main reason for the lower update rate is the limited telemetry data report rate of the autopilot. The CPU load is only about 30 percent because most of the time the filter has to wait for new telemetry data before it proceeds to the next step. Sometimes telemetry data packets are dropped causing longer delays. This has to be investigated further.

7.4 Results

As expected results from the software/hardware-in-the-loop simulations are very similiar to the simulations done previously using the same parameters. No significant differences caused by changes in the environment can be observed.

8 Conclusions

A fast and low cost approach for searching a radio tagged animal using a sensor which gives neither direct bearing nor range information was presented. It is capable of handling the highly nonlinear and non-Gaussian models and by adjusting the number of particles and grid resolution it's possible to adapt it to the computational resources available.



(a) HIL t=440.32 (b) HIL t=832.88

Figure 11: Hardware-In-The-Loop Simulation

9 Future Works

The grid based guidance algorithm sacrifices optimality for real time behaviour. Although the information gain planer has some potential, resource constraints reduce its effectiveness. The search algorithm is only usable with one target at the moment, but it could be easily modified for tags on different frequencies when using one particle set for each target as shown in [Chung and Furukawa, 2009]. Furthermore use of multiple UAVs might be an option to significantly reduce search time, but there is only basic work done on this topic at the moment. The antenna model does not incorporate polarisation as it is assumed that its effect can be neglected in the current setup. This assumption might be wrong and therefore some work should go into this. The next step are flight tests using the real sensor setup. The receiver, antenna and tags are already ordered and the aircraft (a 1:3 scale Piper J-3 Cub which was already used for several UAV missions) is prepared to carry them. The first flight tests shall verify if the model assumptions are correct.

References

[A.G. Longley, 1968] P.L. Rice A.G. Longley. Prediction of tropospheric radio transmission over irregular terrain. Technical report, ERL 79-ITS 67, U.S. Gov. Printing Office, Washington, DC, 1968.

[Arulampalam *et al.*, 2002] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 50(2):174–188, Feb 2002.

[Biotrack, 2009] Ltd Biotrack. Biotrack pip transmitters. Website, June 22, 2009.

[Bourgault *et al.*, 2004] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte. Decentralized bayesian negotiation for cooperative search. volume 3, pages 2681–2686 vol.3, 2004.

[Briner *et al.*, 2003] Thomas Briner, Jean-Pierre Airoldi, Fritz Dellsperger, Simon Eggimann, and Wolfgang Nentwig. A new system for automatic radiotracking of small mammals. *Journal of Mammalogy*, 84(2):571–578, 2003.

[Chen *et al.*, 2007] L. Chen, C. Lee, A. Budhiraja, and R. K. Mehra. PFLib: an object oriented MATLAB toolbox for particle filtering. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 6567, April 2007.

[Chung and Furukawa, 2009] Chern Ferng Chung and Tomonari Furukawa. Coordinated pursuer control using particle filters for autonomous search-and-capture. *Robotics and Autonomous Systems*, 57(6-7):700 – 711, 2009.

[Circuit Design, 2009] Inc. Circuit Design. Radio wave propagation characteristics. Website, June 22, 2009.

[Doucet *et al.*, 2001] Arnaud Doucet, Nando Defreitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, June 2001.

[Furukawa *et al.*, 2007] Tomonari Furukawa, Hugh F. Durrant-Whyte, and Benjamin Lavis. The element-based method - theory and its application to bayesian search and tracking -. In *IROS*, pages 2807–2812. IEEE, 2007.

[Hagn, 1980] G. Hagn. Vhf radio system performance model for predicting communications operational ranges in irregular terrain. *Communications, IEEE Transactions on*, 28(9):1637–1644, Sep 1980.

[ITU, 2001] ITU-R Recommendation P.1546. *Method for Point-to-area Predictions for Terrestrial Services in the Frequency Range 30 MHz to 3000 MHz*, 2001.

[Krten, 2008] Rob Krten. *Getting Started with QNX Neutrino*. QNX Software Systems, 2008.

[Marchetti *et al.*, 2006] Luca Marchetti, Giorgio Grisetti, and Luca Iocchi. A comparative analysis of particle filter based localization methods. In *RoboCup*, pages 442–449, 2006.

[Marchetti *et al.*, 2007] Luca Marchetti, Giorgio Grisetti, and Luca Iocchi. A comparative analysis of particle filter based localization methods. pages 442–449, 2007.

[Ristic *et al.*, 2004] Branko. Ristic, Sanjeev. Arulampalm, and Neil. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, Boston, Ma. ; London :, 2004.

[Struzak, 2004] Ryszard Struzak. Radio-wave propagation basics. eBook, 2004.

[Thrun *et al.*, 2005] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, September 2005.