Contents

Proof of concept experiment

Parallel particle filters
for Tracking of Glimpses of Competing Voices

Joanna Luberadzka
Hendrik Kayser
Volker Hohmann

1 Introduction

1.1 Attentive tracking

Human listeners can follow a desired speaker even in very challenging acoustic conditions with many simultaneously active sound sources. This impressive ability indicates that our auditory system is equipped with a very effective data processing strategies, that allow for extracting useful information about the reality, given raw binaural signal that represents vibrations of the eardrums.

Studies show that the information streaming can be done by sorting the signals by their similarities - parts of the signal with the same pitch, timbre or direction of arrival will be assigned to one stream [9]. However, the mentioned features can continuously change in time, for example the speakers can move in space or change the intonation while speaking.

It has been shown by [3], that following a desired voice can still be successfull, even if there are no constant differences between the sound sources. The study points out that there has to be a mechanism that is responsible for the time integration of the attended stream (movig locus of attention).

1.2 Computational Model

To contribute to understanding the attentive tracking ability, we would like to develop a computational model simulating this phenomenon. Similarly to [3], we use a scenario with two competing talkers. There are no constant differences between the voices (e.g. timbre), but the parameters of the voices change in time. Attentive tracking is simulated by sequential estimation of the parameters of the competing voices based on the features extracted from the binaural signal for each point in time.

1.2.1 Feature Extraction : Glimpses

Feature extraction is in general responsible for redundancy reduction and for preparing the data for further processing steps. There are studies ([**?**],[**?**]) showing that in a multitalker scenario, the cues about a desired target speaker are carried mostly by the so-called glimpses - spectro-temporal pieces of information that are least affected by the background. Removing everything but the glimpses from the signal results in a similar speech intelligibility as when using the whole available information. The most important properties of the glimpses are:

Table 1: Auditory Glimpse Properties

- glimpses are usually **sparse** - they are available only once in a while
- they are **salient** in a sense that they protrude from the background sounds
- they are **robust** in a sense that they provide an uncluttered piece of information about the target
- they have a **single origin**, which means that a glimpse can always be assigned to only one target speaker

We would like the feature extraction stage in the model to simulate the glimpsing that is present in the auditory system. However, it is not trivial to identify the target-related

glimpses without the perfect knowledge about the target and masker energies. We would like to use the periodicity-based glimpse extraction method proposed by [**?**], in which a speech-related glimpse is defined as a spectro-temporal region with high periodic energy.

1.2.2 Sequential State Estimation: Particle Filters

Sequential state estimation is in general responsible for infering the high-level information about the acoustic objects in the scene. One theory about the neural processing states that the predictions about the higher level of abstraction proceeds in a competing hypotheses framework - hypotheses are recursively confronted with a sensory input and updated (Predictive coding [**?**]).

Mathematically it can be described with sequential Bayesian estimation. The objective of the sequential Bayesian estimation methods is to approximate the posterior probability distribution of the current state of a system, given a sequence of observations. This problem is formulated in the following way:

$$p(\vec{y}_n|\vec{x}_{0:n}) = \frac{p(\vec{x}_n|\vec{y}_n)p(\vec{y}_n|\vec{x}_{0:n-1})}{p(\vec{x}_n|\vec{x}_{0:n-1})} \tag{1}$$

$$\textit{Chapman-Kolmogorov equation: } p(\vec{y}_n|\vec{x}_{0:n-1}) = \int p(\vec{y}_n|\vec{y}_{n-1})p(\vec{y}_{n-1}|\vec{x}_{0:n-1})d\vec{y}_{n-1} \tag{2}$$

$$\textit{Normalizing constant: } p(\vec{x}_n|\vec{x}_{0:n-1}) = \int p(\vec{x}_n|\vec{y}_n)p(\vec{y}_n|\vec{x}_{0:n-1})d\vec{y}_n \tag{3}$$

where $\vec{y}$ is a *state vector* characterising the hidden state of the system and $\vec{x}$ is an *observation vector*, containing measured data. $p(\vec{x}_n|\vec{y}_n)$ is the data likelihood distribution - it represents the relationship between the observed data and the hidden state.

$p(\vec{y}_n|\vec{x}_{0:n-1})$ computed in Eq.2 is called the prediction term, since it predicts the future state, given all previous measurements. It is calculated using transition probability $p(\vec{y}_n|\vec{y}_{n-1})$ and the previous estimation result $p(\vec{y}_{n-1}|\vec{x}_{1:n-1})$.

So the estimation at time n can be computed using the estimation at time $n-1$, likelihood $p(\vec{x}_n|\vec{y}_n)$ and transition probability $p(\vec{y}_n|\vec{x}_{n-1})$. Since observation statistics and transition probability distributions are assumed not to change over time, the solution to this problem can be represented as an iteration between the prediction and update step:

Initialize $p(\vec{y}_n|\vec{x}_n) = p(\vec{y}_0)$

for $\boldsymbol{N}$ iterations :

- **Prediction step:** Predict $p(\vec{y}_n|\vec{x}_{0:n-1})$ using $p(\vec{y}_n|\vec{y}_{n-1})$ and $p(\vec{y}_{n-1}|\vec{x}_{1:n-1})$.
- **Update step:** Update posterior $p(\vec{y}_n|\vec{x}_{1:n})$ using predition $p(\vec{y}_n|\vec{x}_{0:n-1})$ and the likelihood evaluated at the current observation $p(\vec{x}_n|\vec{y}_n)$
- n=n+1

end

In each iteration, apart of the previously computed result $p(\vec{y}_{n-1}|\vec{x}_{0:n-1})$, the knowledge of the following probability distributions is required:

- Prior distribution $p(\vec{y}_0)$
- State transition distribution $p(\vec{y}_n|\vec{y}_{n-1})$
- Likelihood distribution $p(\vec{x}_n|\vec{y}_n)$

The procedure above is just a conceptual solution for Bayesian tracking. There is no general analytical solution for Eq.1. Solutions exist only for specific assumptions about the type of the probability distributions. Grid methods might also be used in case the state space is finite and known. If neither the state space nor the distribution type is known, the sampling methods for approximating Eq.1, known as sequential Monte Carlo sampling or particle filtering are used. Since we are dealing with nonlinear relations between the state and observation we want to use particle filtering for the state estimation.

2 Methods

The aim of this experiment is to develop and evaluate a particle filter-based method that will be suitable for tracking voice parameters from the glimpsing features.

Auditory scene consisting of two simultaneously active singing voices is considered. The "singing voice" means a continuously active voice consisting only of voiced excitation (no breaks in the singal, no consonants), which changes pitch and vowels over time. There are no constant differences between the voices. Instead, some of the parameters characterising the voices ($F0$, $F1$, $F2$, α) change over time, following a statistically defined trajectory. We treat these parameters as hidden states of the system, which should be estimated by the particle filter from the observation. Hence, the *state vector* contains the high level information about the acoustic scene like pitch of the contributing voices, their direction of arrival. *Obsvervation vector* simulates information that is available at the higher stages of auditory system - the glimpses at the output of auditory bottom-up processing [2].

To allow for scaling the complexity of the model in a controllable manner, we will in the first place use artificially generated data, instead of multidimensional periodicity-based glimpses proposed by [1]. Artificial observation sequence is generated based on hidden state trajectories, using simple statistical models, introduced later in this document. Nevertheless, the above discussed glimpse properties (See Table 1) are taken into account in the observation generation.

2.1 State trajectory generation

At first, the *ground truth* state trajectory for each voice is generated. Each voice is described by one state vector:

$$\vec{s}_{\imath,n} = \begin{pmatrix} F0_{\imath,n} \\ F1_{\imath,n} \\ F2_{\imath,n} \\ \alpha_{\imath,n} \end{pmatrix}$$

$$\vec{s}_{2,n} = \begin{pmatrix} F0_{2,n} \\ F1_{2,n} \\ F2_{2,n} \\ \alpha_{2,n} \end{pmatrix}$$

The first three dimensions of the state describe fundamental frequency and first two formants in Herz and the fourth dimension describes the direction of arrival in angles.

The state trajectory for voice 1 can be written as

$$\mathcal{T}_{\vec{s}_1} = \{\vec{s}_{1,0}, \vec{s}_{1,1}, ..., \vec{s}_{1,N}\}$$

and the state trajectory for voice 2 can be written as

$$\mathcal{T}_{\vec{s}_2} = \{\vec{s}_{2,0}, \vec{s}_{2,1}, ..., \vec{s}_{2,N}\}$$

, where N is the length of the trajectory.

We assume that the elements of the state are independent from each other.

We want the ground truth trajectories of the state to follow the same statistics, as will be later used in particle filtering. The first element of the trajectory is always drawn from the prior distribution (See Section 3.1):

$$\begin{aligned} \vec{s}_{1,0} &\sim p(\vec{y}_{1,0}) \\ \vec{s}_{2,0} &\sim p(\vec{y}_{2,0}) \end{aligned} \tag{4}$$

Subsequent samples are iteratively drawn from state transition probability distribution (See Section 3.2):

$$\begin{aligned} \vec{s}_{1,n} &\sim p(\vec{y}_{1,n} | \vec{y}_{1,n-1} = \vec{s}_{1,n-1}, \vec{y}_{1,n-2} = \vec{s}_{1,n-2}) \\ \vec{s}_{2,n} &\sim p(\vec{y}_{2,n} | \vec{y}_{2,n-1} = \vec{s}_{2,n-1}, \vec{y}_{2,n-2} = \vec{s}_{2,n-2}) \end{aligned} \tag{5}$$

Since at time $n = 1$ the state element $\vec{s}_{1,n-2}$ (or $\vec{s}_{2,n-2}$) is not yet available, the following distribution is used:

$$\begin{aligned} \vec{s}_{1,1} &\sim p(\vec{y}_{1,n} | \vec{y}_{1,n-1} = \vec{s}_{1,0}, \vec{y}_{1,n-2} = \vec{s}_{1,0}) \\ \vec{s}_{2,1} &\sim p(\vec{y}_{2,n} | \vec{y}_{2,n-1} = \vec{s}_{2,0}, \vec{y}_{2,n-2} = \vec{s}_{2,0}) \end{aligned} \tag{6}$$

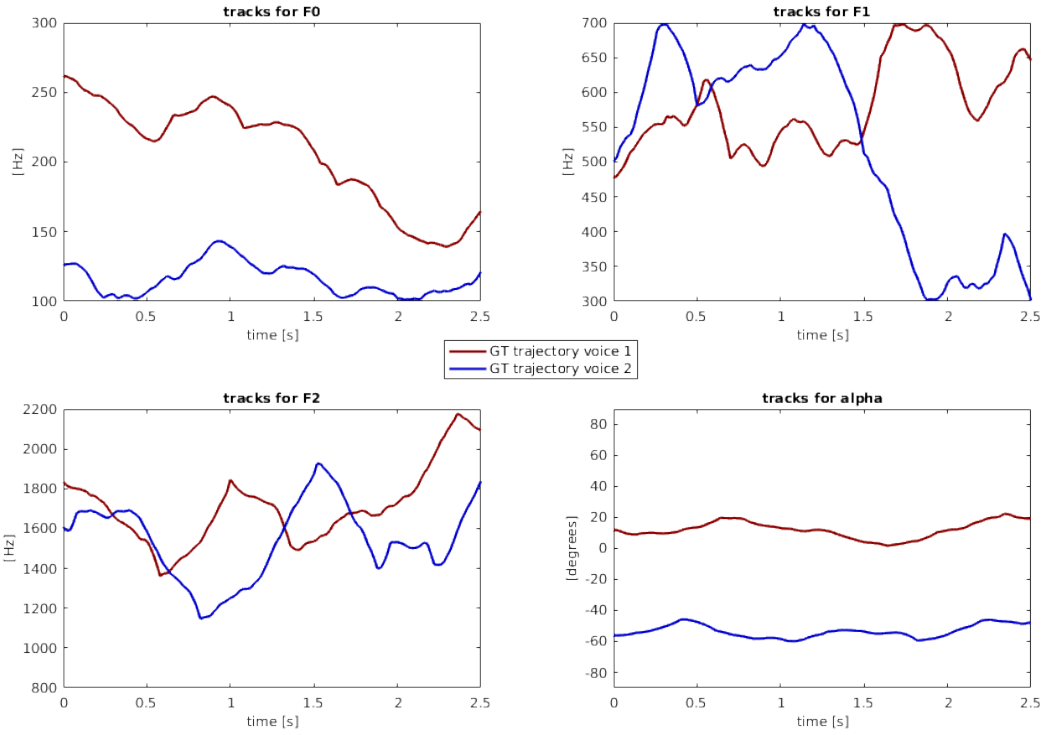

Figure 1: Example of two ground truth state trajectories plotted separately for each dimension.

Fig.1 presents an example of the generated trajectories for two voices in 4 dimensions.

2.2 Observation generation

There are two ground truth state sequences $\mathcal{T}_{\vec{s}_1}$ and $\mathcal{T}_{\vec{s}_2}$, each corresponding to one voice, but only one observation sequence, that bears information about the complete auditory scene:

$$\mathcal{T}_{\vec{o}} = \{\vec{o}_0, \vec{o}_1, ..., \vec{o}_N\}$$

We artificially generate this observation. The artificial observation contains the same quantities as the state element ($F0$, $F1$, $F2$, α):

$$\vec{o}_n = \begin{pmatrix} F0_{\vec{o},n} \\ F1_{\vec{o},n} \\ F2_{\vec{o},n} \\ \alpha_{\vec{o},n} \end{pmatrix}$$

The glimpsing nature of an observation (See Sec. and Table 1) is simulated in the following way: observation at time n can either be empty, it can fully originate from voice 1 or fully originate from voice 2:

$$\vec{o}_n = \begin{cases} \vec{o}_n \sim p(\vec{x}_{1,n}|\vec{y}_{1,n} = \vec{s}_{1,n}) = p(\vec{x}_1|\vec{y}_1 = \vec{s}_{1,n}) & \text{if } z_n \in [0, Q] \\ \vec{o}_n \sim p(\vec{x}_{2,n}|\vec{y}_{2,n} = \vec{s}_{2,n}) = p(\vec{x}_2|\vec{y}_2 = \vec{s}_{2,n}) & \text{if } z_n \in [1-Q, 1] \\ \vec{o}_n = \begin{Bmatrix} NaN \\ NaN \\ NaN \\ NaN \end{Bmatrix} = \overrightarrow{NaN} & \text{if } z_n \in [Q, 1-Q], \end{cases} \tag{7}$$

where z_n is a random variable uniformly distributed in the range $[0, 1]$ (drawn independently for each point in time n), Q is the probability of obtaining a glimpse from voice 1 (which is equal to the probability of obtaining a glimpse from voice 2). Symbol $\sim$ states for *is drawn from a distribution*.

Distribution $p(\vec{x}_{1,n}|\vec{y}_{1,n})$ is the noise model (observation statistics/likelihood) for voice 1 and $p(\vec{x}_{2,n}|\vec{y}_{2,n})$ is the noise model for voice 2. They describe the probability of obtaining data, given a certain state value (See Section 3.3). We are considering a stationary case of the sequential distributions, which means that the data evolves in time, but the distribution from which it is generated remains the same. Therefore we can remove the time index n and instead of $p(\vec{x}_{1,n}|\vec{y}_{1,n})$, write $p(\vec{x}_1|\vec{y}_1)$.

It should be noted that these probability distributions, for each point in time n depend on the $\vec{s}_{1,n}$ or $\vec{s}_{2,n}$ - elements of the ground truth state trajectory $\mathcal{T}_{\vec{s}_1}$ or $\mathcal{T}_{\vec{s}_2}$. So a sequence of observations $\mathcal{T}_{\vec{o}}$ is generated based on the state trajectories of both voices ($\mathcal{T}_{\vec{s}_1}$ and $\mathcal{T}_{\vec{s}_2}$) and a sequence of uniformly distributed random variables $\mathcal{T}_z$. Figure 2 represents

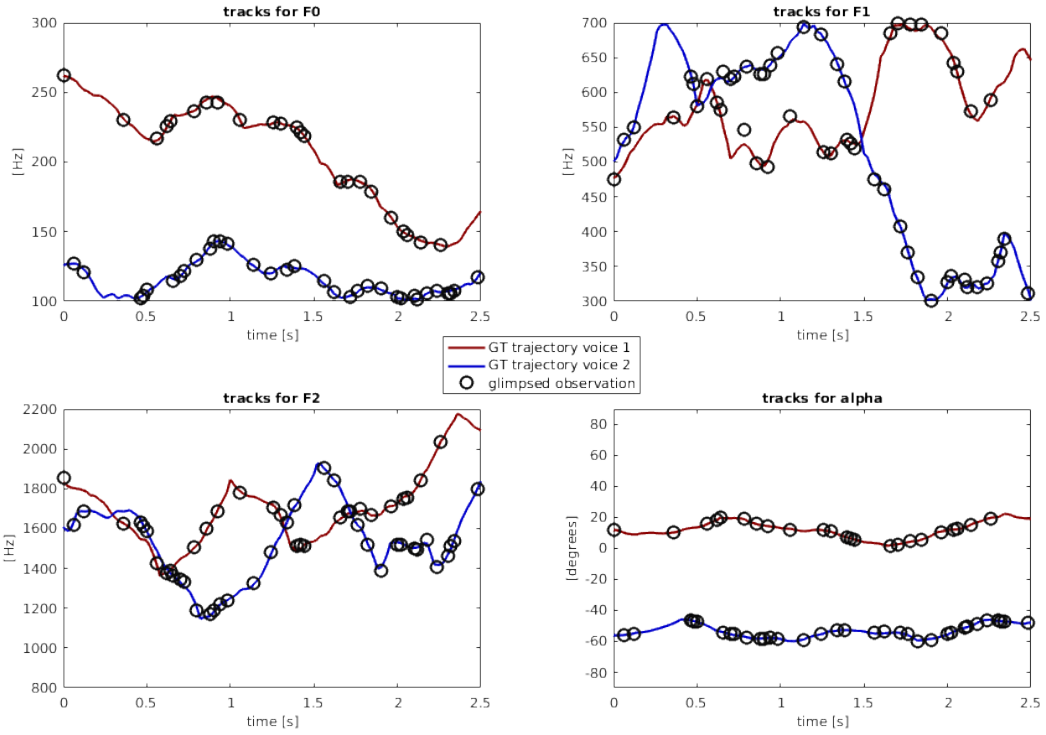

Figure 2: Example of two ground truth state trajectories plotted separately for each dimension.

2.3 State estimation using paralell particle filters

We use a combination of two particle filters $\mathcal{PF}_1$ and $\mathcal{PF}_2$ to estimate the parameters characterising the auditory scene.

The particle filters recursively (for each point in time n) compute the estimates of the state of voices 1 and 2:

$$\hat{\vec{s}}_{1,n}$$

$$\hat{\vec{s}}_{2,n}$$

Later on, the estimation sequences:

$$\mathcal{T}_{\hat{\vec{s}}_1} = \{\hat{\vec{s}}_{1,1}, ..., \hat{\vec{s}}_{1,N}\}$$

and

$$\mathcal{T}_{\hat{\vec{s}}_2} = \{\hat{\vec{s}}_{2,1}, ..., \hat{\vec{s}}_{2,N}\}$$

will be compared with the ground truth sequences $\mathcal{T}_{\vec{s}_1}$ and $\mathcal{T}_{\vec{s}_2}$ (See Section 2.1).

Each particle filter is responsible for estimating the state of one voice.

2.3.1 Initialization

The particle filtering process start with initializing the set $\mathcal{H}$ of K *hypothetical states* of the system - *particles*:

$$\mathcal{H}_{1,0} = \{\vec{h}_{1,0}^{(1)}, ..., \vec{h}_{1,0}^{(K)}\}$$
$$\mathcal{H}_{2,0} = \{\vec{h}_{2,0}^{(1)}, ..., \vec{h}_{2,0}^{(K)}\}$$

and the *weights* assigned to them $\mathcal{W}$:

$$\mathcal{W}_{1,0} = \{w_{1,0}^{(1)}, ..., w_{1,0}^{(K)}\}$$
$$\mathcal{W}_{2,0} = \{w_{2,0}^{(1)}, ..., w_{2,0}^{(K)}\}$$

It is done for each voice separately.

Particles of $\mathcal{PF}_1$ are initialized by drawing K samples from the *state prior* distribution $p(\vec{y}_{1,0})$ and particles of $\mathcal{PF}_2$ are initialized by drawing K samples from the *state prior* distribution $p(\vec{y}_{2,0})$:

$$\begin{aligned} \vec{h}_{1,0}^{(k)} &\sim p(\vec{y}_{1,0}) \ \forall \ k = 1, ..., K \\ \vec{h}_{2,0}^{(k)} &\sim p(\vec{y}_{2,0}) \ \forall \ k = 1, ..., K \end{aligned} \tag{8}$$

(For detailed information about the state prior distribution see Sec.3.1).

In the initialization step, all the weights in the sets $\mathcal{W}_{1,0}$ and $\mathcal{W}_{1,0}$ are given the same value:

$$\begin{aligned} w_{1,0}^{(k)} &= 1/K \ \forall \ k = 1, ..., K \\ w_{2,0}^{(k)} &= 1/K \ \forall \ k = 1, ..., K \end{aligned} \tag{9}$$

2.3.2 State prediction

At the beginning of this step, the time step is incremented

$$n = n + 1.$$

After this operation, the state prediction step modifies the particles from the previous particle sets $\mathcal{H}_{1,n-1}$ and $\mathcal{H}_{2,n-1}$ by adding a system noise component. Each element k of the new particle set $\mathcal{H}_{1,n}$ is drawn from the *state transition* distribution $p(\vec{y}_{1,n}|\vec{y}_{1,n-1}, \vec{y}_{1,n-2})$ (sometimes also called system dynamics):

$$\vec{h}_{1,n}^{(k)} \sim p(\vec{y}_{1,n}|\vec{y}_{1,n-1} = \vec{h}_{1,n-1}^{(k)}, \vec{y}_{1,n-2} = \vec{h}_{1,n-2}^{(k)}) \ \forall \ k = 1, ..., K \tag{10}$$

and correspondingly, for $\mathcal{PF}_2$

$$\vec{h}_{2,n}^{(k)} \sim p(\vec{y}_{2,n}|\vec{y}_{2,n-1} = \vec{h}_{2,n-1}^{(k)}, \vec{y}_{2,n-2} = \vec{h}_{2,n-2}^{(k)}) \ \forall \ k = 1, ..., K$$

Modifying the previous particle set can be seen as predicting what value the current state could get, given that the previous step has a certain value. (For detailed information about the state transition distribution see Sec.3.2).

2.3.3 Glimpse association

If there is a *glimpse* available at time step n, it is assigned to one of the particle filters. The decision which of the particle filters will be fed with the currently available data is made based on the likelihood of the incoming glimpse given the current state estimate:

If $\vec{o}_n$ is a glimpse:

$$\begin{cases} \mathcal{PF}_1 \text{ receives } \vec{o}_n \\ \mathcal{PF}_2 \text{ receives no data} \end{cases} \quad \text{if } \Delta_n > 0$$
$$\begin{cases} \mathcal{PF}_1 \text{ receives no data} \\ \mathcal{PF}_2 \text{ receives } \vec{o}_n \end{cases} \quad \text{if } \Delta_n < 0 \tag{11}$$

The decision parameter Δ is computed in the following way:

$$\Delta_n = \delta_{1,n} - \delta_{2,n} \tag{12}$$

where:

$$\begin{aligned} \delta_{1,n} &= p(\vec{x}_1 = \vec{o}_n | \vec{y}_1 = \hat{\vec{s}}_{1,n-1}) \\ \delta_{2,n} &= p(\vec{x}_2 = \vec{o}_n | \vec{y}_2 = \hat{\vec{s}}_{2,n-1}), \end{aligned} \tag{13}$$

Where $\delta_{1,n}$ is the value of the likelihood distribution for voice 1: $p(\vec{x}_1|\vec{y}_1)$ evaluated at the observation $\vec{o}_n$ and the last state estimation $\hat{\vec{s}}_{1,n-1}$ and $\delta_{2,n}$ is the value of the logarithm of the likelihood distribution for voice 2: $p(\vec{x}_2|\vec{y}_2)$ evaluated at the observation $\vec{o}_n$ and the last state estimation $\hat{\vec{s}}_{2,n-1}$ for that voice (for the computation of the final state estimation see Section 2.3.5)

The decision stage can be summarized now: if at the current time n there is a glimpse available, then by computing Δ it is decided which voice is more likely to produce this observation, given the previous voice state estimate. The particle filter associated with this voice takes th observation as an input. We informally call it the 'winning' particle filter.

The remaining ('losing') particle filter doesn't receive any glimpse and 'freezes' the updating procedure. It means that the weights are equalized [1] and resampling is not done.

Associating a glimpse with one particle filter is based on the assumption that a glimpse is always generated by an individual voice, so only one particle filter should be reacting to that observation, whereas the secod particle filter runs 'freely' - in each iteration it continues to execute the state prediction step, but skips the weight update step.

[1] this is changed in the demo script

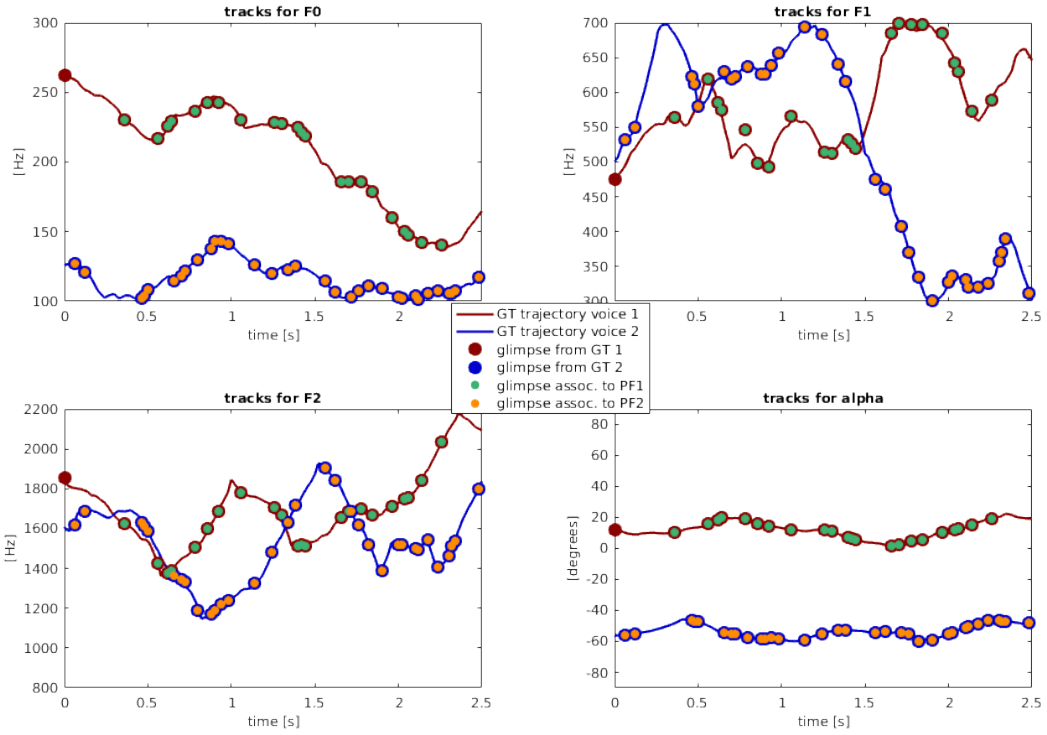

Figure 3: Example of glimpse association, plotted together with the ground truth state trajetories. In this case we can see that there was a robust glimpse association - all glimpses originating from voice 1 were associated with the first particle filter and all glimpses originating from voice 2 were associated with the second particle filter.

2.3.4 Weight update

In this step, currently available data vector $\vec{o}_n$ is used to compute new weights for the particles of the 'winning' particle filter. It can be seen as quantifying the support for each hypothetical state of the system or findig how likely it is that the currently available data point was generated under each of the current hypotheses (particles) in the hypotheses set $\mathcal{H}_1$ or $\mathcal{H}_2$.

This support is computed by *evaluating* the *likelihood* (observation statistics) distribution for each particle given the current data.

For example, if glimpse was associated with $\mathcal{PF}_1$:

$$\begin{aligned} w_{1,n}^{(k)} &= p(\vec{x}_1 = \vec{o}_n | \vec{y}_1 = \vec{h}_{1,n}^{(k)}) \ \forall \ k = 1, ..., K \\ w_{2,n}^{(k)} &= 1/K \ \forall \ k = 1, ..., K \end{aligned} \tag{14}$$

where $p(\vec{x}_1 = \vec{o}_n | \vec{y}_1 = \vec{h}_{1,n}^{(k)})$ is the value of the likelihood probability distribution $p(\vec{x}_1 | \vec{y}_1)$, evaluated at current input $\vec{o}_n$ and particle $\vec{h}_{1,n}^{(k)}$ from the set $\mathcal{H}_{1,n}$.

(For detailed information about the likelihood distribution see Sec.3.3).

2.3.5 Expectation computation

After the update state, weights in sets $\mathcal{W}_{1,n}$ and $\mathcal{W}_{2,n}$ are normalized to 1:

$$\begin{aligned} \tilde{w}_{1,n}^{(k)} &= \frac{w_{1,n}^{(k)}}{\sum_{k'} w_{1,n}^{(k')}} \ \forall \ k = 1, ..., K \\ \tilde{w}_{2,n}^{(k)} &= \frac{w_{2,n}^{(k)}}{\sum_{k'} w_{2,n}^{(k')}} \ \forall \ k = 1, ..., K \end{aligned} \tag{15}$$

Particles in set $\mathcal{H}_{1,n}$ toghether with the normalized weights assigned to them: $\tilde{\mathcal{W}}_{1,n}$ form the approximate *posterior* state distribution $p(\vec{y}_{1,n}|\vec{x}_{1,0:n})$. And correspondingly - particles $\mathcal{H}_{2,n}$ with weights $\tilde{\mathcal{W}}_{2,n}$ form the approximate posterior $p(\vec{y}_{2,n}|\vec{x}_{2,0:n})$.

Finally, the current estimate of the state for a given voice is the expected value of the approximate posterior for that voice:

$$\begin{aligned} \hat{\vec{s}}_{1,n} &= \sum_{k=1}^{K} \tilde{w}_{1,n}^{(k)} \vec{h}_{1,n}^{(k)} \\ \hat{\vec{s}}_{2,n} &= \sum_{k=1}^{K} \tilde{w}_{2,n}^{(k)} \vec{h}_{2,n}^{(k)} \end{aligned} \tag{16}$$

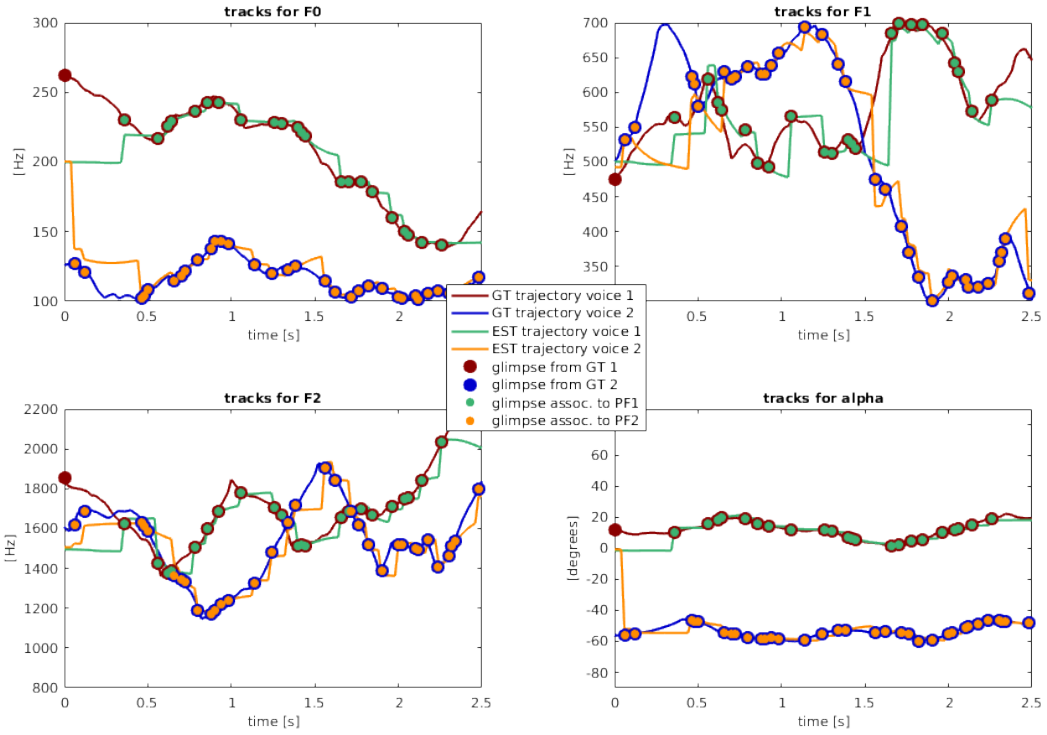

Figure 4: Example of estimated state trajectories, plotted together with the ground truth trajectories and glimpse association. Independent od the glimpse occurance, for each point in time, current states of voice 1 and 2 are estimated as expected value of the particles - this is how we get the estimated state trajectory for each voice.

2.3.6 Resampling

Generally, the resampling step is executed to avoid the sample degeneracy - a very broadly distributed set of particles with have negligible weight. Resampling eliminates particles that have small weights and generated new particles in the proximity of particles with larger weights. It is supposed to increase the precision of sampling in the continuous state space which leads to a better approximation of the posterior state distribution.

The current set of particles $\mathcal{H}_{1,n}$ is overwritten by a new set of particles, according to a used resampling method - function. Only the particle filter, to which a glimpse was assigned performs the resampling step. For example, if glimpse was assigned to $\mathcal{PF}_1$:

$$\begin{aligned} \tilde{\mathcal{H}}_{1,n} &= r(\mathcal{H}_{1,n}) \\ \tilde{\mathcal{H}}_{2,n} &= \mathcal{H}_{2,n} \end{aligned} \tag{17}$$

, where r is a resampling function that transforms the current particle set into a resampled particle set. Different resampling techniques are discussed in Section 4.

3 Probability distributions

In this section, the detailed information about the types of probability distributions used in this experiment will be discussed.

3.1 Prior state probability distribution

The elements of the state vector are considered to be independent, therefore the joint probability can be represented as the product of the probability for each element:

$$\begin{aligned} p(\vec{y}_{\mathtt{1},0}) &= \prod_{i=1}^{D_y} p(y_{\mathtt{1}_i,0}) = \prod_{i=1}^{D_y} \mathcal{U}(y_{\mathtt{1}_i,0}; a_{\mathtt{1}_i}, b_{\mathtt{1}_i}) \\ p(\vec{y}_{\mathtt{2},0}) &= \prod_{i=1}^{D_y} p(y_{\mathtt{2}_i,0}) = \prod_{i=1}^{D_y} \mathcal{U}(y_{\mathtt{2}_i,0}; a_{\mathtt{2}_i}, b_{\mathtt{2}_i}), \end{aligned} \tag{18}$$

where D_y is the number of elements in the state vector, $y_{\mathtt{1}_i,0}$ is the i-th element of the state vector $\vec{y}_{\mathtt{1},0}$ and the notation $\mathcal{U}(y; a, b)$ stands for the univariate uniform distribution of the random variable y over the range $[a, b]$.

The initial range reflects the prior knowledge about where the voice is. In case we have no prior information at all, it will be the same as the range of all the possible values that a given state element can take. (See range $[A_i, B_i]$ in 3.2). The following initial ranges of the speech parameters are used in the experiment:

state el. i	$[a_{\mathtt{1}_i}, b_{\mathtt{1}_i}]$	$a_{\mathtt{2}_i}, b_{\mathtt{2}_i}$
$F0 \in$	$[100, 300]$	$[100, 300]$
$F1 \in$	$[300, 700]$	$[300, 700]$
$F2 \in$	$[800, 2200]$	$[800, 2200]$
$\alpha \in$	$[-90, 90]$	$[-90, 90]$

3.2 State transition probability distribution

We assume that the elements of the state are independent from each other and that the $i-th$ element of the current state vector can only be influenced by the $i-th$ element of the previous state vector. The joint probability can be represented as a following product:

$$\begin{aligned} p(\vec{y}_{\mathtt{1},n} | \vec{y}_{\mathtt{1},n-1}, \vec{y}_{\mathtt{1},n-2}) &= \prod_{i=1}^{D_y} p(y_{\mathtt{1}_i,n} | y_{\mathtt{1}_i,n-1}, y_{\mathtt{1}_i,n-2}) \\ p(\vec{y}_{\mathtt{2},n} | \vec{y}_{\mathtt{2},n-1}, \vec{y}_{\mathtt{2},n-2}) &= \prod_{i=1}^{D_y} p(y_{\mathtt{2}_i,n} | y_{\mathtt{2}_i,n-1}, y_{\mathtt{1}_i,n-2}) \end{aligned} \tag{19}$$

where D_y is the number of elements in the state vector, $y_{\mathtt{1}_i,n}$ is the i-th element of the state vector $\vec{y}_{\mathtt{1},n}$.

The transition model for the i-th element of the state vector for voice $\mathtt{1}$ has the following form:

$$p(y_{\mathtt{1}_i,n}|y_{\mathtt{1}_i,n-1}, y_{\mathtt{1}_i,n-2}) = \begin{cases} \mathcal{N}(y_{\mathtt{1}_i,n}; \tilde{y}_{\mathtt{1}_i,n}, \overset{trans}{\sigma}_{\mathtt{1}_i}), & \text{if } \tilde{y}_{\mathtt{1}_i,n} \in [y_{\mathtt{1}_i,n-1} - 10 \cdot \overset{trans}{\sigma}_{\mathtt{1}_i}, y_{\mathtt{1}_i,n-1} + 10 \cdot \overset{trans}{\sigma}_{\mathtt{1}_i}] \\ & \wedge\ \tilde{y}_{\mathtt{1}_i,n} \in [A_i, B_i] \\ \mathcal{N}(y_{\mathtt{1}_i,n}; y_{\mathtt{1}_i,n-1}, \overset{trans}{\sigma}_{\mathtt{1}_i}) & \text{otherwise} \end{cases} \tag{20}$$

where

$$\tilde{y}_{\mathtt{1}_i,n} = y_{\mathtt{1}_i,n-1} + (y_{\mathtt{1}_i,n-1} - y_{\mathtt{1}_i,n-2})$$

and for voice $\mathtt{2}$ the transition model would then be:

$$p(y_{\mathtt{2}_i,n}|y_{\mathtt{2}_i,n-1}, y_{\mathtt{2}_i,n-2}) = \begin{cases} \mathcal{N}(y_{\mathtt{2}_i,n}; \tilde{y}_{\mathtt{2}_i,n}, \overset{trans}{\sigma}_{\mathtt{2}_i}), & \text{if } \tilde{y}_{\mathtt{2}_i,n} \in [y_{\mathtt{2}_i,n-1} - 10 \cdot \overset{trans}{\sigma}_{\mathtt{2}_i}, y_{\mathtt{2}_i,n-1} + 10 \cdot \overset{trans}{\sigma}_{\mathtt{2}_i}] \\ & \wedge\ \tilde{y}_{\mathtt{2}_i,n} \in [A_i, B_i] \\ \mathcal{N}(y_{\mathtt{2}_i,n}; y_{\mathtt{2}_i,n-1}, \overset{trans}{\sigma}_{\mathtt{2}_i}) & \text{otherwise} \end{cases} \tag{21}$$

where

$$\tilde{y}_{\mathtt{2}_i,n} = y_{\mathtt{2}_i,n-1} + (y_{\mathtt{2}_i,n-1} - y_{\mathtt{2}_i,n-2})$$

and the notation $\mathcal{N}(x; \mu, \sigma)$ used above stands for the univariate gaussian distribution of the random variable x with mean μ and standard deviation σ.

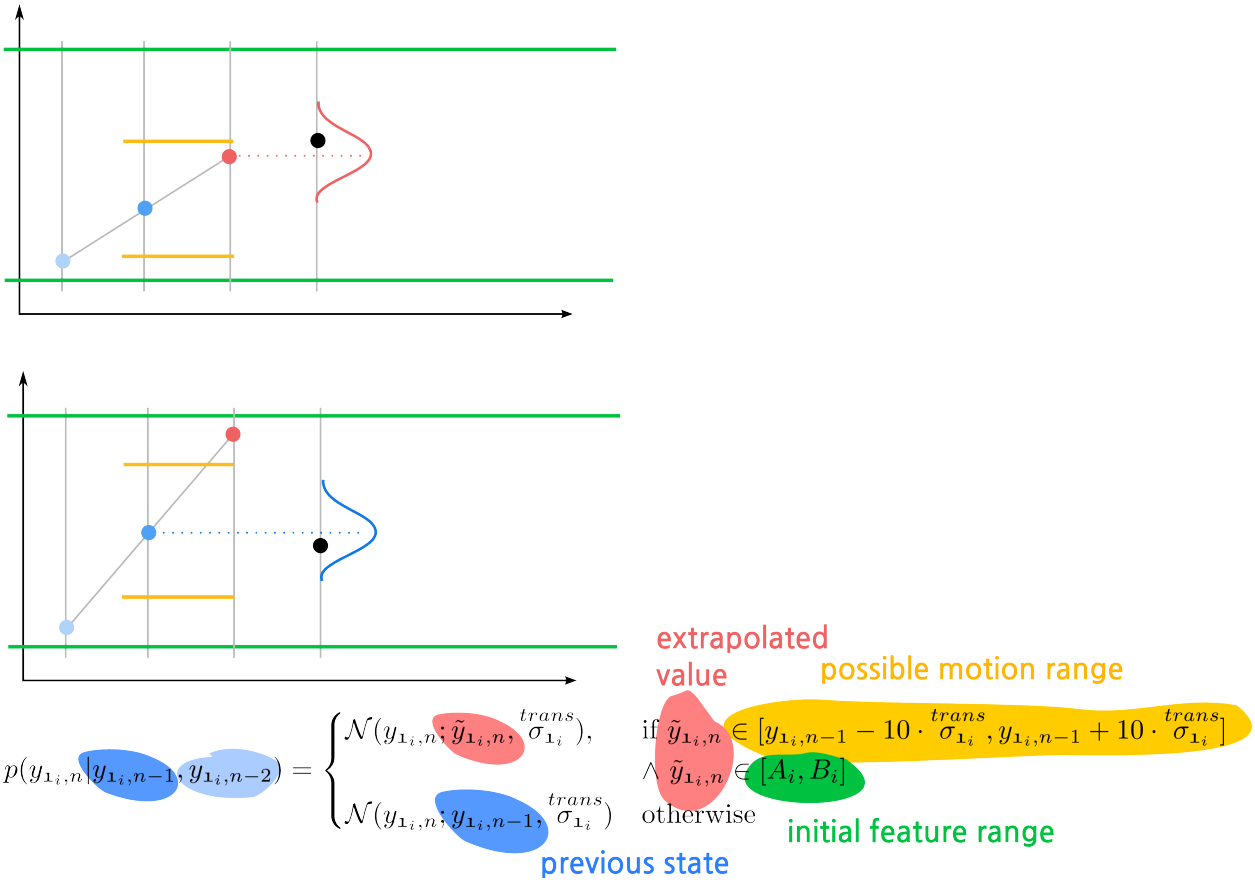

Figure 5: State prediction for the particle k (See Section 2.3.2)can be summarized with these steps (here notation for voice $\mathtt{1}$): a)For each dimension i of the particle, extrapolate $\tilde{h}^{(k)}_{\mathtt{1}_i,n}$ based on the difference between k-th particle at last two time steps :$h^{(k)}_{\mathtt{1}_i,n-2}$ and $h^{(k)}_{\mathtt{1}_i,n-1}$ b) If the extrapolated value $\tilde{h}^{(k)}_{\mathtt{1}_i,n}$ deviates from the previous particle $h^{(k)}_{i,n-1}$ more than $10 \cdot \sigma_i$ or if $\tilde{h}_{\mathtt{1}_i,n}$ is not not in the range of possible values of that state element $[A_i, B_i]$, then the extrapolation is not done. c) The new state value is drawn from a Gaussian pdf with standard deviation $\overset{trans}{\sigma}_{\mathtt{1}_i}$, centered either at extrapolated state $\tilde{h}_{\mathtt{1}_i,n}$ (if extrapolation was done) or at the previous state value $h_{\mathtt{1}_i,n-1}$ (in case the extrapolated value was not in the appropriate range).

For each state element the Gaussian has a defined standard deviation:

state el. i	$\overset{trans}{\sigma}_{\mathtt{1}_i}$	$\overset{trans}{\sigma}_{\mathtt{2}_i}$
$F0$	1	1
$F1$	5	5
$F2$	10	10
α	0.25	0.25

The state elements have the following ranges of the possible values (the same for both voices):

state el. i	$[A_i, B_i]$
$F0 \in$	$[100, 300]$
$F1 \in$	$[300, 700]$
$F2 \in$	$[800, 2200]$
$\alpha \in$	$[-90, 90]$

Figure 5 visualizes the usage of the state transition probability in the state prediction step of the particle filter.

3.3 Likelihood probability distribution

We consider the elements of the state and observation vector to be conditionally independent. The joint probability is then a product of the individual probability distributions:

$$\begin{aligned} p(\vec{x}_{\mathtt{1},n}|\vec{y}_{\mathtt{1},n}) = p(\vec{x}_{\mathtt{1}}|\vec{y}_{\mathtt{1}}) = \prod_{j=1}^{D_x} p(x_{\mathtt{1}_j}|\vec{y}_{\mathtt{1}}) \\ p(\vec{x}_{\mathtt{2},n}|\vec{y}_{\mathtt{2},n}) = p(\vec{x}_{\mathtt{1}}|\vec{y}_{\mathtt{1}}) = \prod_{j=1}^{D_x} p(x_{\mathtt{2}_j}|\vec{y}_{\mathtt{2}}), \end{aligned} \tag{22}$$

where D_x is the dimensionality of the observation vector and $x_{\mathtt{1}_j}$ is the $j-th$ element of the observation vector for voice $\mathtt{1}$: $\vec{x}_{\mathtt{1},n}$ We are considering a stationary case of the sequential distributions, which means that the data evolves in time, but the distribution from which it is generated remains the same. Therefore we can remove the time index n. We also know that the $j-th$ element of the observation vector only depends on one element ($j-th$ element) of the state vector:

$$\begin{aligned} p(\vec{x}_{\mathtt{1}}|\vec{y}_{\mathtt{1}}) = \prod_{j=1}^{D_x} p(x_{\mathtt{1}_j}|y_{\mathtt{1}_j}) \\ p(\vec{x}_{\mathtt{2}}|\vec{y}_{\mathtt{2}}) = \prod_{j=1}^{D_x} p(x_{\mathtt{2}_j}|y_{\mathtt{2}_j}), \end{aligned} \tag{23}$$

And the individual probability distribution of the $j-th$ element has the following form:

$$\begin{aligned} p(x_{\mathtt{1}_j}|y_{\mathtt{1}_j}) = \mathcal{N}(x_{\mathtt{1}_j}; y_{\mathtt{1}_j}, \overset{obs}{\sigma}_{\mathtt{1}_i}) \\ p(x_{\mathtt{2}_j}|y_{\mathtt{2}_j}) = \mathcal{N}(x_{\mathtt{1}_j}; y_{\mathtt{2}_j}, \overset{obs}{\sigma}_{\mathtt{2}_i}), \end{aligned} \tag{24}$$

where the notation $\mathcal{N}(x; \mu, \sigma)$ used above stands for the univariate gaussian distribution of the random variable x with mean μ and standard deviation σ.

For each state element the Gaussian has a defined standard deviation:

state el. i	$\overset{obs}{\sigma}_{1_i}$	$\overset{obs}{\sigma}_{2_i}$
$F0$	1	1
$F1$	5	5
$F2$	10	10
α	0.25	0.25

We can see that the observation is just a noisy version of the state vector: a gaussian noise component is added to the state value in each dimension.

4 Resampling techniques

The goal of the particle filtering is to approximate a continuous multidimensional probability distribution by a discrete distribution formed by a limited number of particles. Therefore it is crucial to distribute the particles over the state space so that they can provide the best approximation. Resampling is done to avoid sample degeneracy, which is a result of particles being distributed too widely. However, on the other side, there is an inverse problem called sample impoverishement, which can be seen as particles being over concentrated. It occurs when very few particles have significant weight while most other particles with small weights are excluded during the resampling process. There is a trade-off of the need to focus on the meaningful particles and the need to maintain the diversity of particles.

A resampling method should best fit the data. We would like to assess which of the below presented resampling techniques best fits the sparse, but robust *glimpses*.

4.1 Standard resampling

In the standard resampling procedure, the new particle set is drawn from a current discrete approximation of the posterior distribution:

$$\tilde{\vec{h}}_n^{(k)} \sim p(\vec{y}_n|\vec{x}_{0:n}). \tag{25}$$

4.2 Effective particles resampling

Too frequent resampling can lead to the problem of sample impoverishment. To avoid it we can execute the resampling only when we know that the particles are degenerated. A commonly used measure of degeneracy is the effective sample size:

$$\hat{N}_{eff} = \frac{1}{\sum_{k=1}^{K} w_n^{(k)}} \tag{26}$$

if $\hat{N}_{eff} < K/10$, then the resampling is executed.

4.3 Naive kernel resampling

One reason for the sample impoverishment problem is the fact that we are sampling from a discrete probability distribution. If the transition noise component added in the prediction

step is not big enough, the particles will always be concentrated around a very narrow region of a state space.

To deal with this, instead of sampling directly from a discrete distribution, we can fit a continuous probability distribution to the available particles-weights pairs and sample from the obtained distribtion to form new particle set. However, probability distribution fitting is a subject of research on its own and the fitting method used in resampling should not cause the explosion of the computational effort.

The method that we use - *naive kernel* computes expected value and variance of the particles in each dimension and then draws new particles from a multivariate gaussian with a diagonal covariance matrix.

4.4 Meanshift resampling

A different approach to the problem of sample impoverishment is to use observed data in the resampling procedure - to perform the data-driven resampling. For the limited system noise, it makes sense to direct particles in the direction of the observation, so that the region, the next observation can possibly fall within is well represented by the particles.

In the simulation study we use the *meanshift* resampling, in which all the particles are shifted by the same value in the direction of the observation:

$$\begin{aligned}
&\tilde{\vec{h}}_n^{(k)} = \tilde{\vec{h}}_n^{(k)} - (\hat{\vec{s}}_n - \breve{\vec{s}}_n), \text{ where} \\
&\hat{\vec{s}}_n = \sum_{k=1}^{K} \tilde{w}_n^{(k)} \vec{h}_n^{(k)} \text{ is the expected value and} \\
&\breve{\vec{s}} = G(\vec{o}_n) \text{ is the candidate state}
\end{aligned} \tag{27}$$

G being a function that generates a potential/candidate state from the observation. In this case the candidate state is sampled from a Gaussian centered at the observation $\vec{o}_n$.

Steering the particles in the direction of the current observation can be done under the assumption that the incoming data is reliable. This is especially adequate in case of glimpses - they might be rarely available, but always provide a robust information about the system.

4.5 Combined resampling

Combined resampling combines the *naive kernel* and *meanshift* - first the particles are shifted in the direction of the observation and then a gaussian kernel is build arround the shifted particles.

5 Simulation experiment

5.1 Experiment data

We tested a total number of 420 conditions, each being a combination of the following parameters:

- Resampling techniques:
  - *No resampling*
  - *Standard resampling*
  - *Effective particles resampling*
  - *Naive kernel resamplings*
  - *Meanshift resampling*
  - *Combined resampling*
- Probability of glimpse occurance:
  - $Q = 0.05, 0.1, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5$
- Factor F, by which the standard deviation of the observation noise is multiplied ($\overset{used}{\sigma} = F * \overset{obs}{\sigma}$):
  - $F = 1, 3, 5, 10, 15, 30, 50$

Fig.6 presents a sample data sequence with a varying glimpse probability Q and Fig.8 presents a sample data sequence with varying variance of the observation noise.

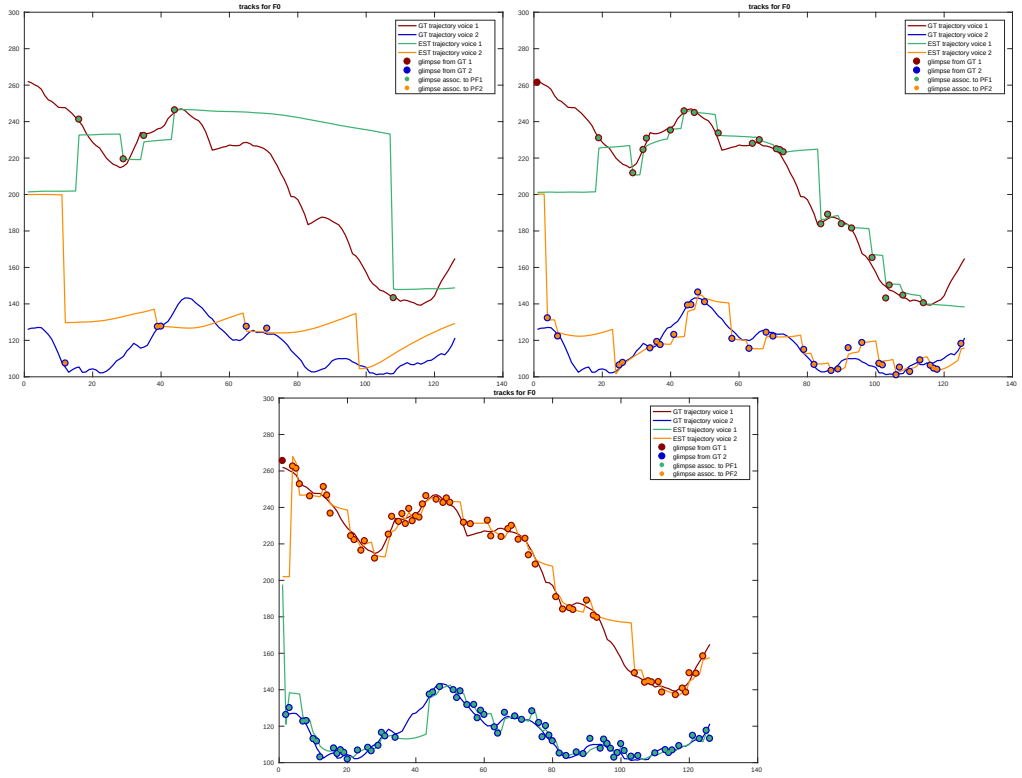

Figure 6: Figure presents an example of simulation data (in one dimension) for different probability of glimpse occurance Q. In the top plot $Q = 0.05$, in the middle plot $Q = 0.3$ and in the bottom plot $Q = 0.5$.

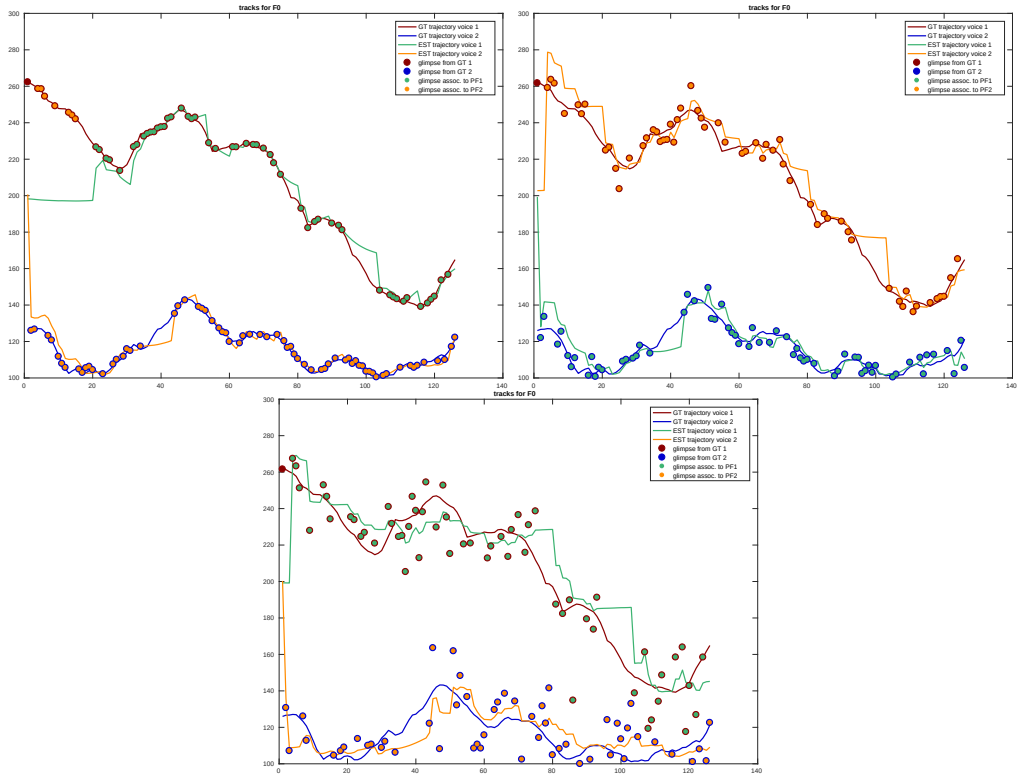

Figure 7: Figure presents an example of simulation data (in one dimension) for different variance of the observation noise $\overset{obs}{\sigma}$. In the top plot $\overset{obs}{\sigma} = 1$, in the middle plot $\overset{obs}{\sigma} = 3$ and in the bottom plot $\overset{obs}{\sigma} = 5$. In all dimensions (which are not shown here, $\overset{obs}{\sigma}$ is multiplied by the same factor.)

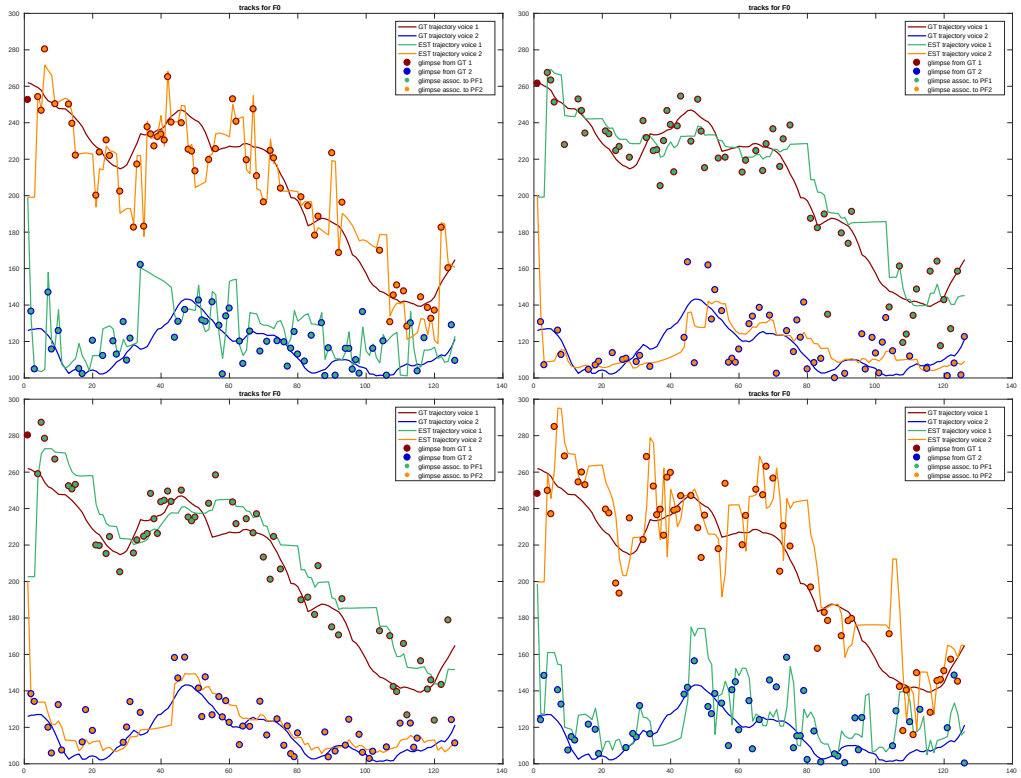

Figure 8: Figure presents an example of simulation data (in one dimension) for different resampling techniques. Starting from the left right corner: *no resampling, standard resampling, naive kernel resampling, meanshift resampling.* It can be seen that different resampling techniques result in different estimation sequences.)

We generate $L = 1000$ *trials*, consisting two ground truth trajectories of length $N = 126$, which correspond to a signal of $2.5s$ length for the data aquisition rate of $50Hz$ (which is used for synthesizing the signals later on). Each condition is tested with the same sample trajectories (trials). In all conditions we used a number of particles $K = 1000$.

5.2 Performance Measures

5.2.1 Track Assignment Procedure

To quantify the performance of the particle filter for each resampling technique, the estimated sequences:

$$\mathcal{T}_{\hat{\vec{s}}_1} = \{\hat{\vec{s}}_{1,0}, \hat{\vec{s}}_{1,1}, ..., \hat{\vec{s}}_{1,N}\}$$

and

$$\mathcal{T}_{\hat{\vec{s}}_2} = \{\hat{\vec{s}}_{2,0}, \hat{\vec{s}}_{2,1}, ..., \hat{\vec{s}}_{2,N}\}$$

have to be compared with the ground truth sequences $\mathcal{T}_{\vec{s}_1}$ and $\mathcal{T}_{\vec{s}_2}$ for each trial.

It has to be decided which of the estimated tracks ($\mathcal{T}_{\hat{\vec{s}}_1}$ or $\mathcal{T}_{\hat{\vec{s}}_2}$) is an estimate of the ground truth track $\mathcal{T}_{\vec{s}_1}$ and which is the estimate of the ground truth track $\mathcal{T}_{\vec{s}_2}$.

Based on glimpse association matrix: G-assignment First method for assigning the tracks uses the ground truth information about the origin of the glimpses that are presented to the system. It is based on the glimpse association matrix G:

$$G = \begin{pmatrix} g_{1,1} & g_{1,2} \\ g_{2,1} & g_{2,2} \end{pmatrix} \tag{28}$$

s

where

$g_{j,k}$ is the percentage of glimpses belonging to track j, that were associated with track k.

We compute the sum of both diagonals of the matrix G and the one that results in a bigger value indicates the final track assignment. For example, if $g_{1,1} + g_{2,2} < g_{2,1} + g_{1,2}$ then we consider track $\mathcal{T}_{\hat{\vec{s}}_1}$ an estimate of a ground truth track $\mathcal{T}_{\hat{\vec{s}}_2}$, and track $\mathcal{T}_{\hat{\vec{s}}_2}$ estimate of the a ground truth track $\mathcal{T}_{\hat{\vec{s}}_1}$.

Based on track distance matrix : D-assignment We would like ot be able to quantify the system performance also if we do not have information about the real origin of the glimpses. In such case we have to assign tracks based on the track distance matrix D:

$$D = \begin{pmatrix} d_{1,1} & d_{1,2} \\ d_{2,1} & d_{2,2} \end{pmatrix} \tag{29}$$

where

$d_{j,k}$ is the root mean square error (RMSE) between ground truth trajectory $\mathcal{T}_{\vec{s}_j}$ and estimated trajectory $\mathcal{T}_{\hat{\vec{s}}_k}$:

$$d_{j,k} = \sqrt{\frac{\sum_{n=1}^{N} \left(||\vec{s}_{j,n} - \hat{\vec{s}}_{k,n}|| \right)^2}{N}} \tag{30}$$

Similarly to the first method, we compute the sum of both diagonals of the matrix D, but this time the one that results in a smaller value indicates the final track assignment. For example, if $d_{1,1} + d_{2,2} > d_{2,1} + d_{1,2}$ then we consider track $\mathcal{T}_{\hat{\vec{s}}_1}$ an estimate of a ground truth track $\mathcal{T}_{\vec{s}_2}$, and track $\mathcal{T}_{\hat{\vec{s}}_2}$ estimate of the a ground truth track $\mathcal{T}_{\vec{s}_1}$.

5.2.2 Track Assignment Performance

It might happen that one (or both) estimated tracks cannot be unambiguously assigned to one of the ground truth tracks (See Fig.9). We would like to identify those cases and compute the percentage of the unambiguous track assignements across 1000 trials. We use either glimpse association matrix G or the track distance matrix D (depending on which method was used to assign tracks (See Sec. 5.2.1)). To identify ambiguous estimations, we compute the sum of rows of the matrix (G or D). For matrix G (See Sec.5.2.1), if any sum of the rows is bigger than both sum of diagonals, it means that the track assignement is ambiguous. For matrix D, (See Sec.5.2.1) if any sum of the rows is smaller than both sums of diagonals, it means that the track assignement is ambiguous.

Track Assignment Performance (TAP) is the percentage of the unambiguous assignments across 1000 trials.

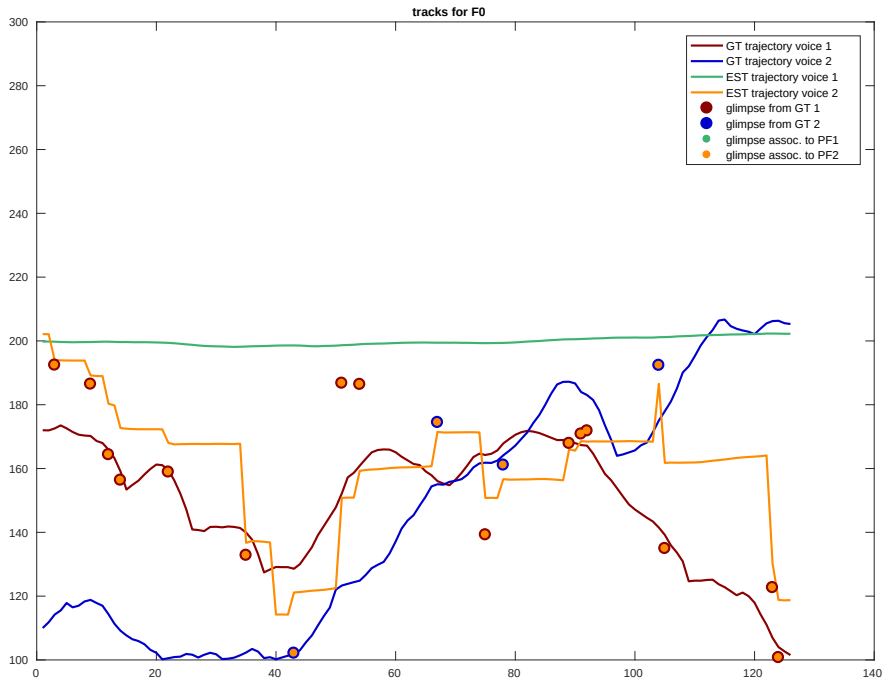

Figure 9: Figure presents an example of the ambiguous estimations in one dimension - it is not clear if the estimation track should be assigned to ground truth sequence $\mathcal{T}_{\vec{s}_1}$ or $\mathcal{T}_{\vec{s}_2}$.

5.2.3 Glimpse Association Performance

Glimpse Association Performace (GAP) is the measure that tells us how many of all glimpses presented to the system were correctly associated. To compute this measure, we first need to already have done the track assignment (either based on D or based on G). GAP is computed as the mean of the elements on the diagonal of G matrix, that corresponds to the track assignement (either $g_{1,1}$ and $g_{2,2}$ or $g_{2,1}$ and $g_{1,2}$)

5.3 RMSE in each dimension ($RMSE_i$)

Having assigned the estimation tracks to the ground truth tracks, we would like to quantify the similarity between the sequences. The most straightforward measure of similiarity is the root mean squared error. For dimension i of the state vector:

$$RMSE_i = \sqrt{\frac{\sum_{n=1}^{N}(s_{1_i,n} - \hat{s}_{1_i,n})^2}{N}} \tag{31}$$

5.3.1 Normalized total RMSE ($RMSE_{total}$)

To compute the overall error for the whole system, we have to compute the error based on normalized tracks. To do that, we first compute the normalized tracks using the ranges of possible values in each dimension:

$$\begin{aligned} &\mathcal{T}_{\vec{s}_1}^{(norm)} \\ &\mathcal{T}_{\vec{s}_2}^{(norm)} \\ &\mathcal{T}_{\hat{\vec{s}}_1}^{(norm)} \\ &\mathcal{T}_{\hat{\vec{s}}_2}^{(norm)} \end{aligned} \tag{32}$$

And using them we compute the total normalized root mean square error ($RMSE_{total}$):

$$RMSE_{total} = \sqrt{\frac{\sum_{n=1}^{N}(\vec{s}_{1,n}^{(norm)} - \hat{\vec{s}}_{1_i,n}^{(norm)})^2}{N}} \tag{33}$$

6 Results

Looking at the results of the simulation (Fig. -) the following effects can be observed:

- The higher the glimpse probability Q the better the performance of the system.
- The higher the observation noise (factor F), the worse the performance of the system.
- For low noise - data-driven resampling is the best choice.
- For high noise - data-driven resampling is the worst choise (kernel resampling the best).
- For very low observation noise *no resampling* is much worse than any resampling technique.
- Although *no resampling* results in a high RMSE, the GAP and TAP is high. Although the estimation tracks are very noisy, they fluctuate around the ground truth tracks.

- There are no significant differences between *meanshift* and *combined* resampling - combination doesn't lead to any additional improvement.
- *naive kernel* resampling is slightly better than *standard* and *effective particles* resampling, but since the difference is not very big, we can say that it belongs to the same category of methods.
- for many measures - there has to be sufficient Q
- many correctly identified glimpses (most conditions above 70
- up to observation noise with factor F track assignment is very good

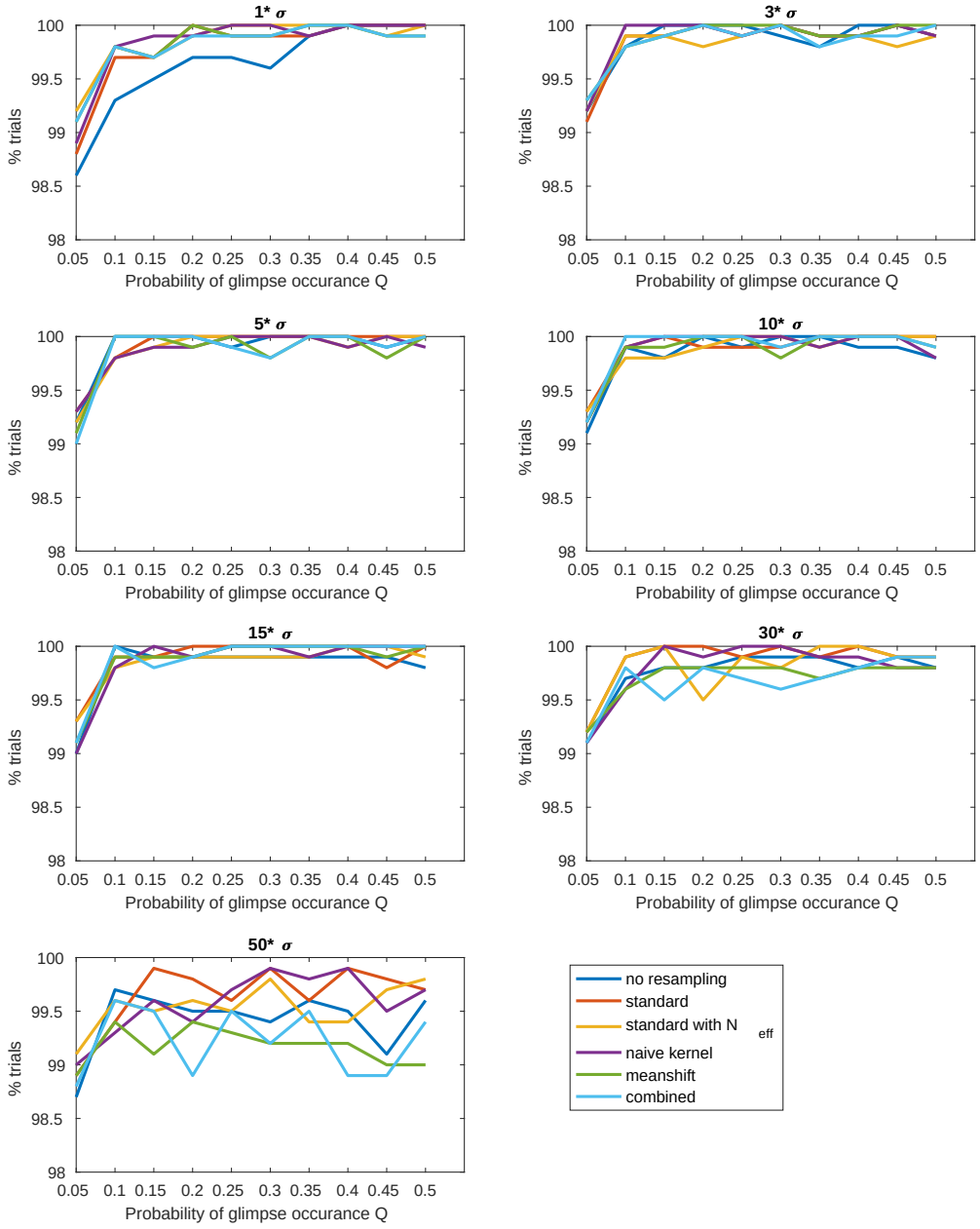

Figure 10: **TAP using G-assignment.** Figure presents the TAP as a function of glimpse probability Q for different resampling techniques (depicted as different colors) and for varying amount of observation noise.

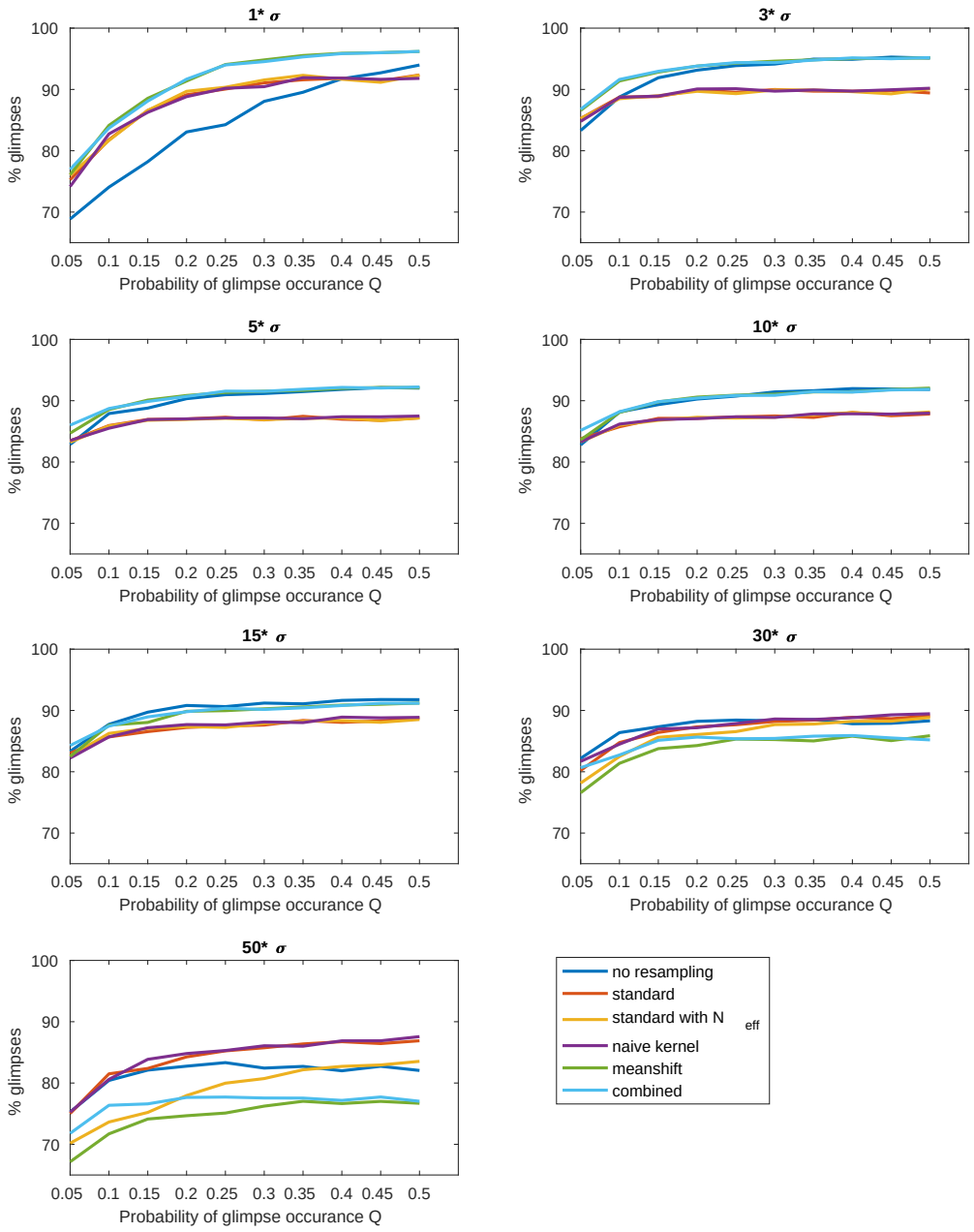

Figure 11: **TAP using D-assignment.** Figure presents TAP computed based on glimpse association matrix D as a function of glimpse probability Q for different resampling techniques (depicted as different colors) and for varying amount of observation noise.

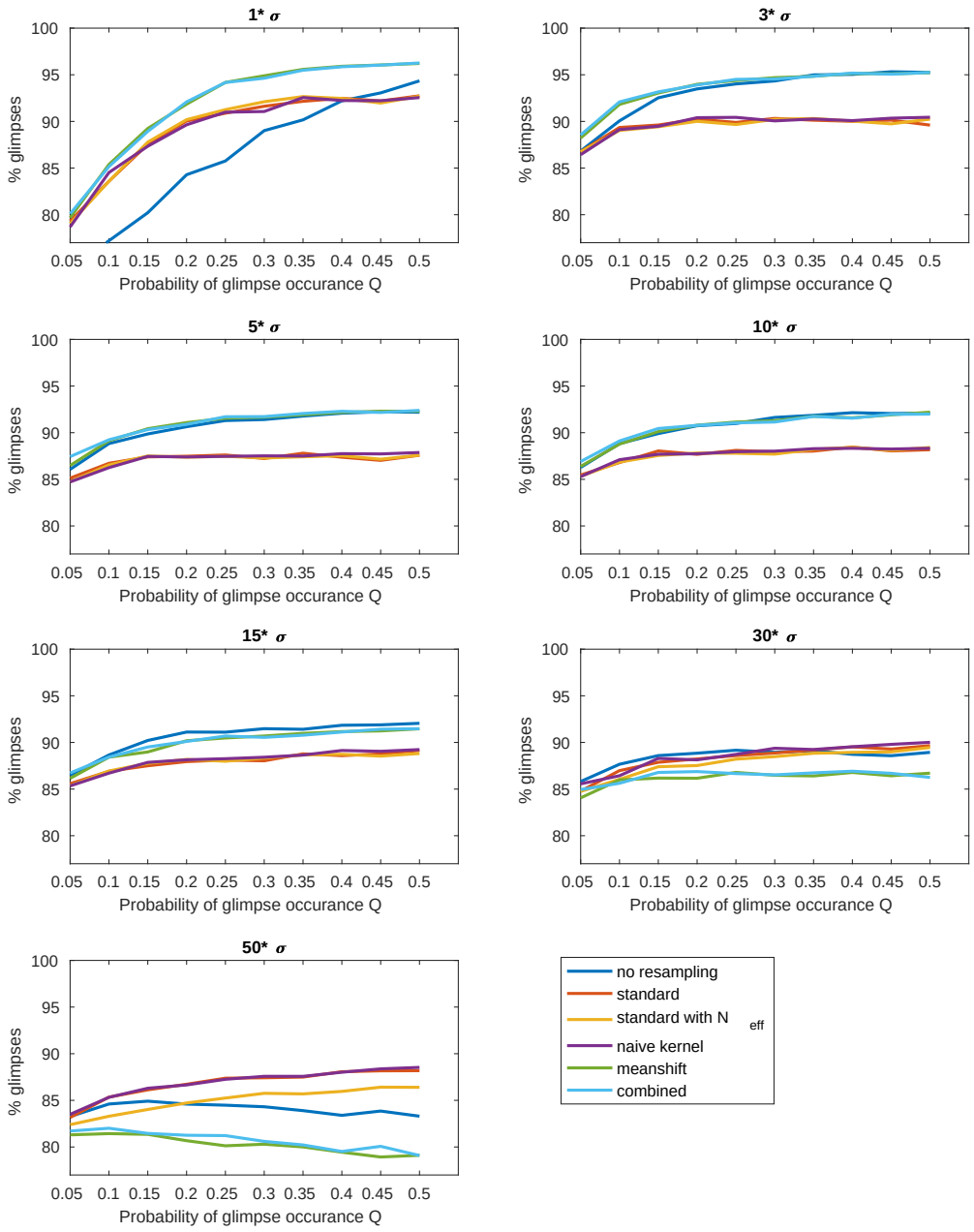

Figure 12: **GAP using G-assignment.** Figure presents GAP computed based on glimpse association matrix G as a function of glimpse probability Q for different resampling techniques (depicted as different colors) and for varying amount of observation noise.

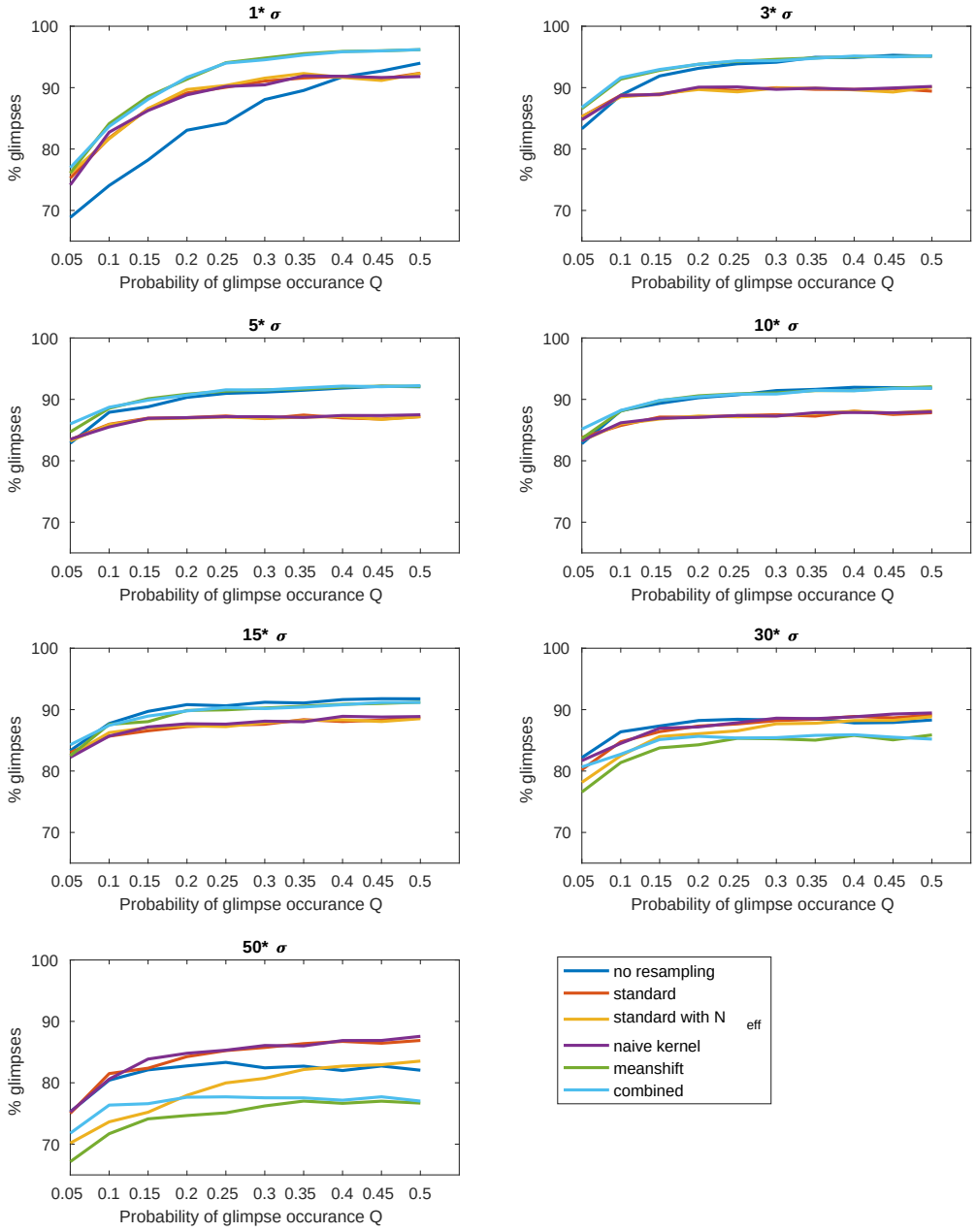

Figure 13: **GAP using D-assignment.** Figure presents GAP computed based on track distance matrix D as a function of glimpse probability Q for different resampling techniques (depicted as different colors) and for varying amount of observation noise.

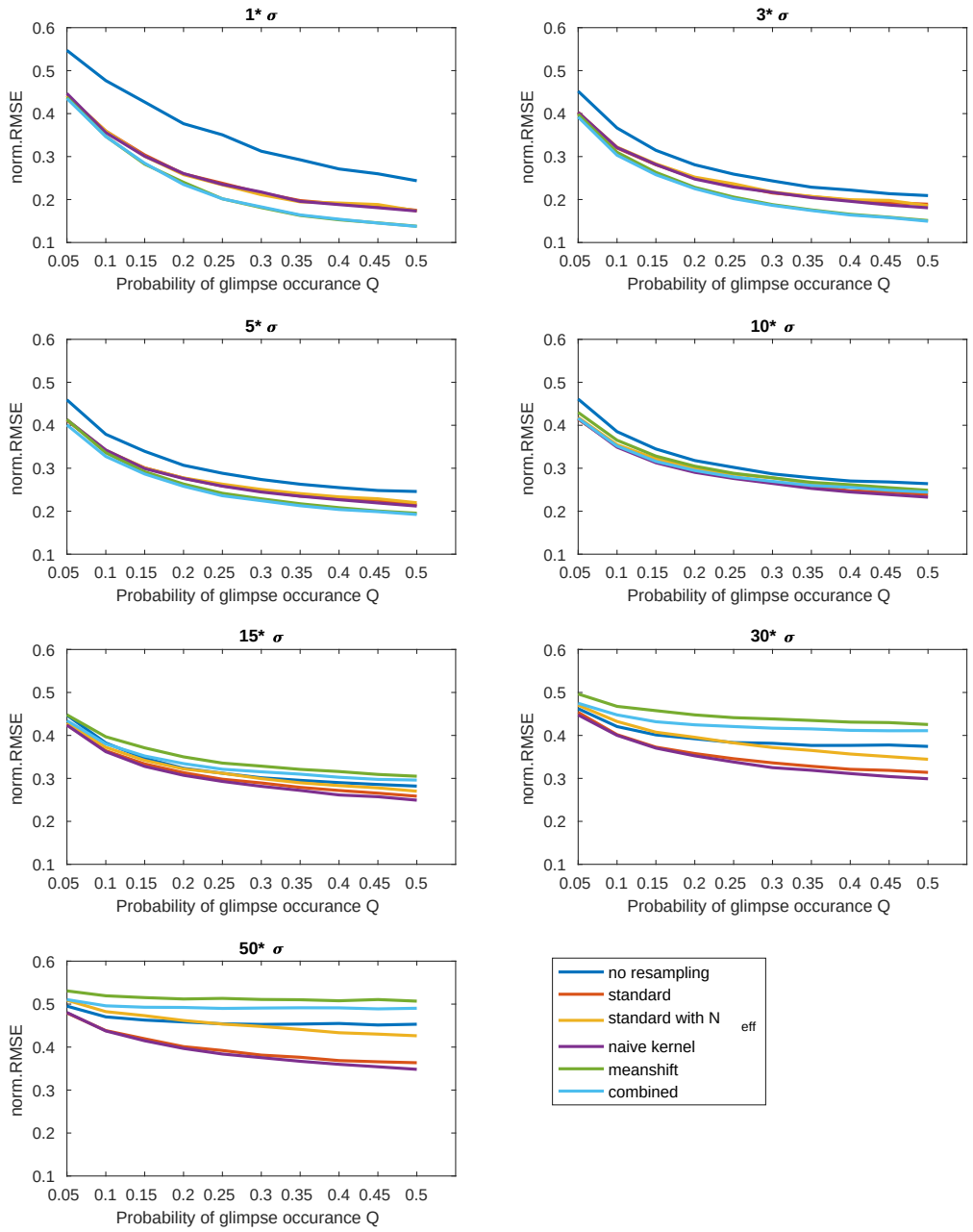

Figure 14: ***RMSE*$_{total}$ using G-assignment.** Figure presents total normalized RMSE between ground truth and estimated tracks (assigned according to glimpse association matrix G) for different resampling techniques (depicted as different colors) and for varying amount of observation noise.

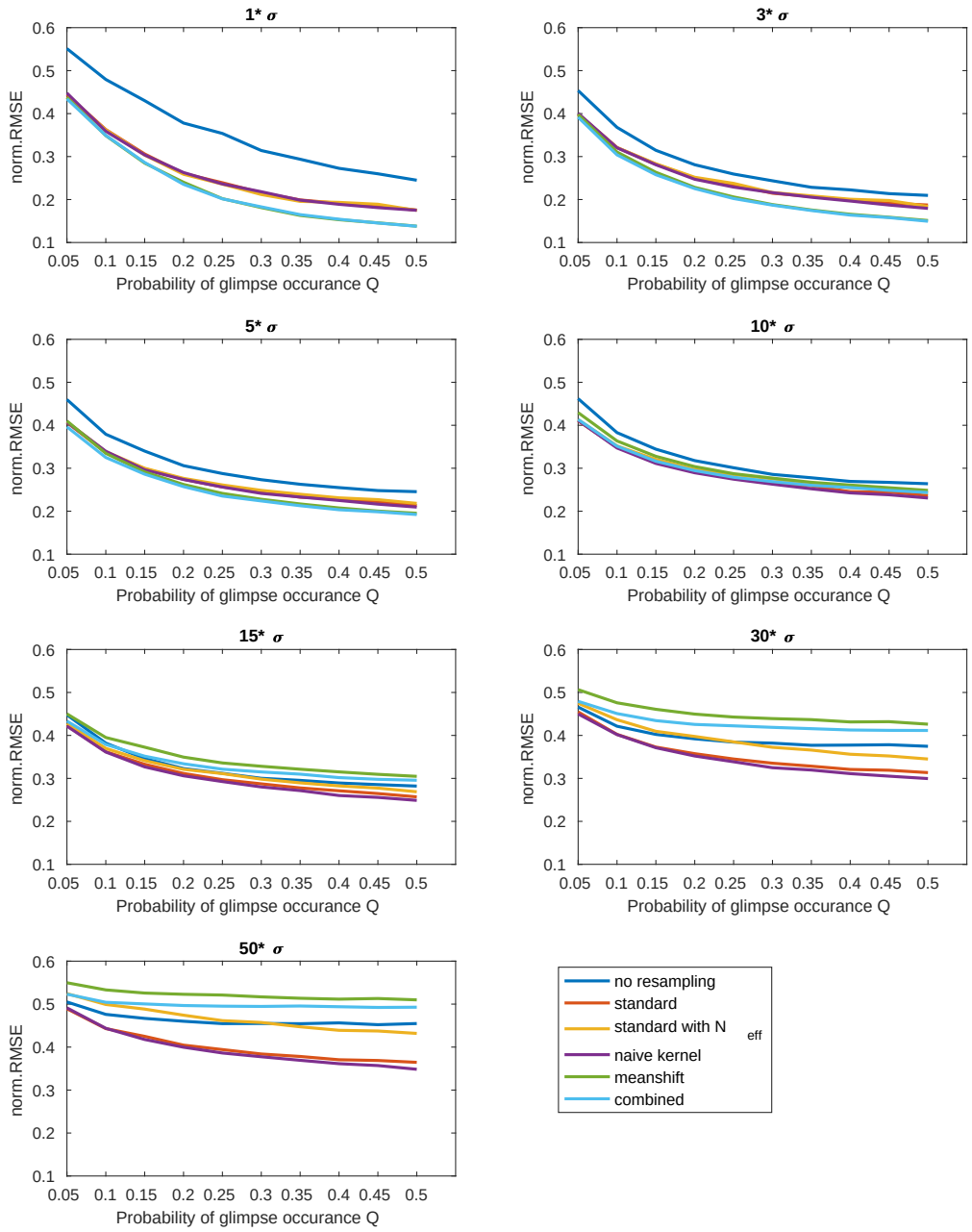

Figure 15: $\boldsymbol{RMSE_{norm}}$ **using D-assignment.** Figure presents total normalized RMSE between ground truth and estimated tracks (assigned according to glimpse association matrix D) for different resampling techniques (depicted as different colors) and for varying amount of observation noise.

7 Conclusions and Discussion

The following conclusions can be made based on the obtained results:

- In general, the paralell particle filters with glimpse association can be used for tracking of two simultaneous events in the 4-dimensional state space using sparsely occuring data.

- Tracking performance is not bad above the glimpse probability of $Q = 1$. It varies with the frequency of receiving glimpses.

  Looking at the individual trials, it mostly depends on the proximity of trajectories (especially at the beginning) if the estimation is successful or not. In the psychoacoustic study [3] they also show that the voices can be distinguished only if the trajetories are far enough from ach other. Here we generated trajectories completely separately and we didn't limit them anyhow.

  The resulting estimated tracks were also not smoothed, so even systematic fluctuations where contributing to the RMSE. It should

- Discrimination performance is very good. In most cases, even if the glimpse probability is quite small, we can differentiate between the trajectories. For modeling a discrimination task, as the one in [3] it should be sufficient.

- Resampling vs Noise: it might be beneficial to adjust resampling technique to the amount of noise. We know about the glimpses that they contain a robust information, so we expect the observation noise of the glimpses to be very small. Then it makes sense to use the data driven resampling.

- There is a mismatch between the theoretical requirements for the system dynamics and the proposed state transition probability distribution. We do not fulfill the first-order markov process property of the system, which is assumed in case of particle filtering - in the state transition we always take two previous states into account.

References

[1] Josupeit, Angela, and Volker Hohmann. "Modeling speech localization, talker identification, and word recognition in a multi-talker setting." *The Journal of the Acoustical Society of America* 142.1 (2017): 35-54.

[2] A. Josupeit, Kopco, N., and Hohmann, V., "Modeling of speech localization in a multi-talker mixture using periodicity and energy-based auditory features" *J. Acoust. Soc. Am.*, vol. 139, no. 5. pp. 2911-2923, 2016.

[3] Kevin J.P. Woods, Josh H. McDermott "Attentive Tracking of Sound Sources". *Current Biology* 25, 1-9 ,August 31, 2015

[4] Nix, Johannes, and Volker Hohmann. "Combined estimation of spectral envelopes and sound source direction of concurrent voices by multidimensional statistical filtering." *IEEE transactions on audio, speech, and language processing* 15.3 (2007): 995-1008.

[5] Bishop, C. "Pattern Recognition and Machine Learning" *Information Science and Statistics* 1st edn. 2006. corr. 2nd printing edn. Springer, New York (2007).

[6] Bishop, Christopher M., Markus Svensen, and Goeffrey E. Hinton. "Distinguishing text from graphics in on-line handwritten ink" ,*Frontiers in Handwriting Recognition* 2004. IWFHR-9 2004. Ninth International Workshop on. IEEE, 2004.

[7] Husmeier, Dirk "Modelling conditional probability densities with neural networks" Diss. University of London, 1997.

[8] Dennis H. Klatt "Software for a cascade/parallel formant synthesizer" *The Journal of the Acoustical Society of America* 67, 971 (1980)

[9] Darwin, C. J. "Listening to speech in the presence of other sounds." *Philosophical Transactions of the Royal Society of London B: Biological Sciences 363.1493 (2008): 1011-1021.*

[10] Li, Tiancheng, et al. "Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches." *Expert Systems with applications* 41.8 (2014): 3944-3954.

[11] Arulampalam, M. Sanjeev, et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking." *IEEE Transactions on signal processing* 50.2 (2002): 174-188.