ROBOT PARTS' REARRANGEMENT - SENSOR UNCERTAINTY REDUCTION USING PARTICLE FILTERS

by

Haluk BAYRAM

B.S. in Control and Computer Engineering, Erciyes University, 2002

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in System and Control Engineering

Boğaziçi University

2006

ACKNOWLEDGEMENTS

I acknowledge with gratitude the invaluable assistance, support and time given in the preparation of this thesis by my advisors Prof. Işıl Bozma and Prof. Ayşın Ertüzün. I thank them for their patience and encouragement that carried me on through difficult times, and for their insights and suggestions that helped to shape my research skills. Their valuable feedback contributed greatly to this thesis. I am thankful to Deniz Gençağa for the discussions on particle filtering theory.

I would like as well to thank to my thesis committee members Prof. Yorgo Istefanopulos, Prof. Levent Akın and Prof. Hakan Deliç.

This work has been supported in part by Bogazici University Scientific Research Projects (BAP) Grant BAP05A202 and State Funding Agency (DPT) Grant 03K120250. I would like to thank TUBITAK for an academic scholarship grant during my master education.

Last but not least, I thank my loving wife, my parents for always being there when I needed them most, and for supporting me.

ABSTRACT

ROBOT PARTS' REARRANGEMENT - SENSOR UNCERTAINTY REDUCTION USING PARTICLE FILTERS

This thesis adresses the parts' moving problem under noisy sensory information. In this scenario, a 2D workspace contains an actuated robot and a set of unactuated parts. The discrepancy between the robot's and/or the parts' real and measured positions may lead to jerky movements or even collisions in the parts' moving problem we are concerned with. In contrast to previous work, sensory data is no longer assumed to be perfect. Hence the robot needs to approximate state information, taking its highly nonlinear nature into account. It accomplishes this using particle filters, which implement a recursive Bayesian filter in nonlinear and/or nongaussian environments. For the model of parts which turns out to be linear, the approach reduces to Kalman filtering. First the robot's dynamic model and the measurement model are modified to incorporate the inaccuracies in the sensory data; and then the particle filter is utilized to get improved positional estimates. Enhancements in the robot's movements and reduction in the number of collisions have been verified through extensive computer simulations. An evaluation of its theoretical performance is presented based on the Cramer-Rao lower bound. Finally, a series of experiments with EDAR provide insight into real-time performance.

ÖZET

ROBOT PARÇA TAŞIMA PROBLEMİNDE ALGILAYICI BELİRSİZLİĞİNİN PARÇACIK SÜZGEÇLER İLE AZALTIMI

Bu tezde, gürültülü algısal veri durumunda parça taşıma problemi ele alınmıştır. Bu senaryoda, iki boyutlu çalışma ortamında hareket edebilen bir robot ve hareket yeteneği olmayan parçalar vardır. Robot ve parçaların gerçek ile ölçülen konumları arasındaki fark sarsıntılı harekete veya çarpışmalara yol açabilmektedir. Önceki çalışmadan farklı olarak, algısal verinin artık tam doğru olmadığı kabul edilmektedir. Dolayısıyla robotun durum bilgisine yüksek doğrusal olmayan yapısını dikkate alarak yaklaşıklaması gerekmektedir. Bu parçacık süzgeçler kullanılmasıyla başarılmıştır. Parçacık süzgeçler doğrusal olmayan ve/veya Gauss olmayan ortamlarda Bayes süzgecini yinelemeli olarak gerçekler. Parçaların modeli doğrusal olduğundan, yaklaşım Kalman süzegece çevrilir. İlk olarak, robotun dinamik ve ölçüm modeli belirsizlikle birleştirilir. Sonra, parçacık süzgeç kullanılarak konumsal kestirimde iyileştirme yapılır. Robotun hareketindeki iyileşmeler ve çarpışma sayısındaki azalma kapsamlı bilgisayar benzetimleri ile doğrulanmıştır. Başarımın kuramsal değerlendirmesi Cramer-Rao alt sınırı kullanılarak yapılmıştır. EDAR ile yapılan deneyler sistemin gerçek zaman başarımına ışık tutmuştur.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

$\|.\|$	Norm of a vector
$b \in \Re^{2p}$	State vector of all the parts
$b_i \in \Re^2$	Position vector of part i
$\overline{b}_i \in \Re^{2(p-1)}$	$\{b_1, ..., b_{i-1}, b_{i+1}, ..., b_p\}$
$D_x\psi$	Gradient of ψ with respect to x
$\mathbb{E}\{.\}$	Expectation operator
$e_i \in \Re^p$	The ith orthonormal basis vector
f	System transition function
$g \in \Re^{2p}$	Vector of all the goals
$g_i \in \Re^2$	Goal vector of part i
h	Measurement function
I_n	Identity matrix in $\Re^n$
$k \in \mathcal{Z}^+$	Artificial potential function design parameter
m	Index of the next part
p	Number of the parts
P	Index set of the parts $P = \{1, \cdots, p\}$
$p(.)$	Probability density function
$q(.)$	Proposal density
$r \in \Re^2$	state vector of the robot
$r_a \in \Re^2 \times SO(1)$	Augmented state vector of the robot $[r\ \theta]^T$
r_k^i	Particle i at time k
$\hat{r}_k$	Estimate of the state at time k
$\Re$	Set of real numbers
$SO(1)$	Configuration space of planar rotation
w_k^i	Weight of particle i at time k
z_k	Measurement at time k
$\mathcal{Z}^+$	Set of positive integers
β	Obstacle function

γ	Goal function
δ	Delta-Dirac function
η	Dynamical noise
ν	Measurement noise
$\rho_i \in \Re^+$	Radius of part i
Σ	Covariance matrix
ϕ	Artificial potential function for *next-part*
φ	Artificial potential function for *mate-part* subtask
ψ	Artificial potential function for *move-part* subtask

2D	Two dimensional
ASIR	Auxiliary sampling importance resampling
comp	Task complexity measure
CRLB	Cramer-Rao lower bound
i.i.d.	Independently and identically distributed
MC	Monte Carlo
npl	Normalized part path length measure
nrl	Normalized robot path length measure
pdf	Probability density function
PF	Particle filter
pi	Positional inaccuracy measure
ppee	Part positional estimation error measure
RMS	Root Mean Square
rpee	Robot positional estimation error measure
SIR	Sampling importance resampling
SIS	Sequential importance resampling
SMC	Sequential Monte Carlo

1. INTRODUCTION

This thesis is concerned with a geometrically simplified version of warehouseman's problem [1] under sensor uncertainty. The problem setup is as follows: The workspace contains a disk-shaped robot which has two degrees of freedom motion and a manipulator with grasping and holding capabilities. The workspace also contains a set of disk-like parts which are unactuated and thus cannot move by themselves. The robot's task is to move all the parts to their prespecified destination locations while having no collisions along the way. As there is no guarantee that the parts will be left undisturbed, the robot is required to employ a reactive strategy. A feedback-based event-driven approach based on artificial potential functions has been presented in [2], assuming that the robot (i) has perfect (online) knowledge of the locations of the parts and own position, and (ii) has perfect (online) knowledge of its joint positions. Since sensory measurements are fed back from the optical encoders and camera-based vision system which are both prone to sensor inaccuracies, these assumptions may not hold in real-time implementations. The discrepancy between the robots' actual and measured positions may cause uneven movements and even possibly collision. Since the robot does not have accurate positional information about itself and the parts, even in cases where there is no collision, the parts' positioning accuracy may deteriorate due to sensing inaccuracies. Hence, the robot cannot perform its task purely based on raw sensory data, but now needs to estimate positional information from noisy measurements.

1.1. Problem Statement

The workspace contains a robot and a set of disk shaped parts with varying radii. The robot's task is to move all the parts from their initial arbitrary configuration to the specified goal configuration with no collisions without any a priori plan in a completely event-driven manner. All the positions including that of the robot and the parts are a priori unknown and are sensed instantaneously from a camera-based vision system. However, due to measurement inaccuracy, the sensed data is noisy. The robot should react to the sensed arrangement of parts based on this inaccurate data. As it cannot

rely entirely on the measurements, the problem is how to alleviate the sensor noise for an improved estimate of the positions of the robot and of all the parts in order to ensure successful task completion.

1.2. Related Literature

The related work includes two distinct areas: Reactive Parts' Rearrangement problems from robotics and Particle Filters from signal processing and uncertainty reduction.

1.2.1. Reactive Parts' Rearrangement

One approach to achieving reactivity has been based on artificial potential functions [3, 4]. However, their usage has been limited since most constructions suffer from undesired local minima and thus are not ensured of convergence to the desired goal positions. A sequel of work reported has then been able to demonstrate that it is possible to overcome this shortcoming through carefully constructed functions. Navigation functions have been shown to have global convergence properties and a construction with this property for the case of robot navigation among static obstacles has been presented in[5]. The approach has then been generalized to linear parts' moving or rearrangement problems where it has been shown that by sequentially switching among a family of feedback controllers, we can generate a plan that is ensured of convergence or termination [6]. An actual implementation in 2D, as presented in [2] has demonstrated the robustness of the approach in case of dynamically changing environments. However, in all previously reported work, sensory information has been perfect.

1.2.2. Sensor Inaccuracy

Robustness against sensor noise is an important issue in most robotic applications. In this case, the robot should not rely completely on the measurements coming from its sensors. Cox and Wilfong state that "Using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with

autonomous capabilities" [7]. The robot needs to deal with the uncertainty in the information provided by the measurements. One of the key developments in robotics is the adoption of probabilistic techniques. Probabilistic robotics integrates imperfect models and imperfect sensors through probabilistic laws, such as Bayes' rule. Particle filter (PF) has been proposed as a probabilistic approach for estimating the state of the dynamic system – for the case of highly nonlinear problems in [8, 9]. The PF depending on probabilistic principles is among the most promising candidates to providing a comprehensive and real-time solution to reliable position estimation. Several variants of the PF such as sampling importance resampling (SIR), auxiliary sampling importance resampling (ASIR), and Rao-Blackwellized PF have then been introduced within a generic framework of the sequential importance sampling (SIS) algorithms [10]. Their application in robotic problems has been studied by many researchers where Bayesian filtering with particle-based density representations are used to localize a mobile robot. Their advantage stems from the fact that they solve numerically the associated optimal filtering problem, which in general does not admit a closed-form solution [11].

Dellaert *et al.* have introduced the Monte Carlo Localization method [12]. By using Monte Carlo (MC) type methods, they combine the advantages of grid-based Markov localization with the efficiency and accuracy of Kalman filter based techniques and use the approach in applications involving laser and the sonar measurements. It should be noted that PFs are used for positioning based on cellular phone measurements, for integrated navigation in aircraft, and for target tracking in aircraft and cars in [13]. A probabilistic algorithm for simultaneously estimating the position of a mobile robot and the positions of nearby people in a previously mapped environment is developed in [14]. This approach, called the conditional PF, tracks a large distribution of people locations conditioned upon a smaller distribution of robot poses over time. Real-time PFs which make use of all sensor information even when the filter update rate is below the update rate of the sensors have been reported [15]. Some of this work have also investigated the theoretical bounds of performance. In particular, Cramer-Rao lower bound is used for evaluation of performance of PF in [16, 10, 17]. The bound can be used to provide information on the fundamental performance level that can be reached for the estimation.

1.3. Contribution of the Thesis

The contributions of this thesis can be summarized as follows:

- The more realistic case of sensor inaccuracy in parts' rearrangement tasks is considered and the previously proposed feedback-based strategy based on artificial potential functions is integrated with particle filters in an iterative manner as to enable continuous update of the state estimates,
- Cramer-Rao lower bound of the resulting system is computed and compared with the actual performance obtained via an extensive simulation study.
- A series of experiments with EDAR is conducted to evaluate its performance in a real-time application.

1.4. Approach

Our approach is based on integration of PFs with feedback-based event driven approach.

- First, the robot's dynamic model is modified to incorporate noisy sensor measurements.
- Next, PFs are constructed in order to get an improved positional estimate for the robot and the parts.
- The robot's position is updated using the probability density function (pdf) thus constructed.
- As the parts are unactuated, PFs simplify out to be Kalman filters which are then used to improve the estimate of the parts' positions.

1.5. Outline of the Thesis

The thesis is organized in the following chapters. The first chapter presents an introduction to the thesis including problem statement, related literature and a summary of the proposed approach. The feedback-based event-driven approach to

the parts' rearrangement problem is presented in Chapter 2. Chapter 3 presents the overview of PF, its application to the rearrangement problem, and the computation of the Cramer-Rao lower bound. In Chapter 4, an extensive set of simulations with 3 and 6 parts rearrangement tasks are presented along with quantitative measures of performance including estimation error with respect to the Cramer-Rao lower bound. Chapter 5 presents results with EDAR - a parts' mover robot operating purely in a feedback-based manner.

2. FEEDBACK-BASED EVENT-DRIVEN PARTS MOVING

The parts' moving problem is defined by a robot that can move freely, grip a part and carry it from a location to another location and a set of disk shaped parts that are arbitrarily located and need to be moved from their arbitrary initial placements to their goal positions. Each is mathematically represented following the formalism of [2]: The robot is parametrized by its radius $\rho_r \in \mathcal{Z}^+$ and is located at $r \in \Re^2$. Its gripper makes an angle $\theta \in SO(1)$ with the x axis. The augmented robot state $r_a \in \Re^2 \times SO(1)$ is defined as $r_a = [r\ \theta]^T$. The set of parts is denoted by $P = \{1, \cdots, p\}$, $p \in \mathcal{Z}^+$. Each part $i \in P$ is again parametrized by its radius $\rho_i \in \Re$. Current and goal locations are defined by $b_i \in \Re^2$ and $g_i \in \Re^2$ respectively. The state of all the parts is a vector $b \in \Re^{2p}$ defined as $b = \sum_{i \in P} b_i \otimes e_i$, where $e_i \in \Re^p$ are the unit vectors in $\Re^p$. The goal vector of all the parts is $g \in \Re^{2p}$ defined as $g = \sum_{i \in P} g_i \otimes e_i$.

In feedback-based event-driven approach, the robot control is modelled as an hybrid system that operates purely on state feedback. In this perspective, the robot implicitly decomposes its overall task into a continuous sequence of subtasks of moving each part respectively via a switching mechanism as depicted in Figure 2.1 [18]. It should be noted that there is no a priori explicit task decomposition as is typically the case in planning based approaches. Accordingly, the robot has the following rules:

- One subtask gets to be executed at a time.
- Subtasks are competing and the robot selects the subtask based on an urgency measure as defined by *next-part* function.
- For each part, a subtask consists of sequence of two stages: i) *Mate-part*: The robot moves to mate with part, ii) *Move-part*: In case of successful mating, the robot moves the part to its goal position.

This switching mechanism is invoked repetitiously in a reactive manner until all the parts are moved to their goal positions by the robot.

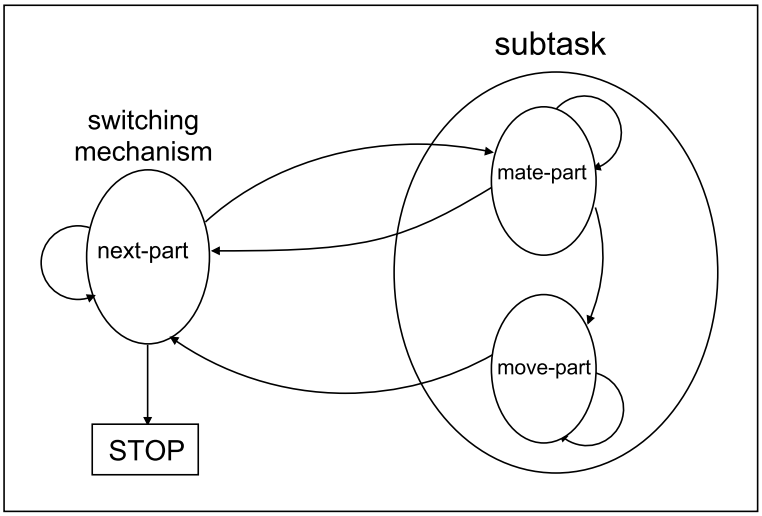

Figure 2.1. Parts' moving game

The robot chooses one among the competing subtasks using an index valued *next-part* function $m(b) : \Re^{2p} \rightarrow P$ [2]:

$$m(b) = \arg\max_{i \in P} \left\| \left(I_2 \otimes e_i^T \right) D_b \phi(b) \right\| \tag{2.1}$$

This function picks out the component of b, whose direction of descent on $\phi(b)$ is the steepest. The function $\phi(b) : \Re^{2p} \rightarrow \Re$ is defined as $\phi(b) = \left(\gamma^{k_1}(b) / \beta(b) \right)$ where $k_1 \in \mathcal{Z}^+$. The term $\gamma(b) : \Re^{2p} \rightarrow \Re$ is defined as $\gamma(b) = \|b - g\|^2$ while the denominator $\beta(b) : \Re^{2p} \rightarrow \Re$ denotes the obstacle function of the pairwise part positions defined as $\beta(b) = \prod_{i \in P} \prod_{j \in P}^{j>i} \|b_i - b_j\|^2 - (\rho_i - \rho_j)^2$.

2.1. Robot Part Mating

The mating control laws are defined by a collection of smooth scalar valued maps $\varphi_i(r, b) : \Re^p \times \Re^{2p} \rightarrow \Re$, $\forall i \in P$. Each φ_i is defined as [2]:

$$\varphi_i(r, b) = \frac{\gamma_i^{k_2}(r, b)}{\beta(r, b)} \tag{2.2}$$

where $\gamma_i(r, b) : \Re^2 \times \Re^{2p} \rightarrow \Re$ is the squared Euclidian distance between the robot and part i defined as $\gamma_i(r, b) = \|r - b_i\|^2$, $\forall i \in P$. The obstacle function $\beta(r, b) : \Re^2 \times \Re^{2p} \rightarrow \Re$ is defined as $\beta(r, b) = \prod_{j \in P} \|r - b_j\|^2 - (\rho_r - \rho_j)^2$. The constant

$k_2 \in \mathcal{Z}^+$ is a positive integer chosen appropriately. The robot's motion toward part i is governed by the dynamical system:

$$\dot{r} = -D_r\varphi_i(r, b) \tag{2.3}$$

where $D_r\varphi_i$ is the gradient of φ_i with respect to r.

2.2. Robot Part Moving

Once the robot mates with part i, the robot-part coupled structure moves as a single body in the extended space $\Re^2 \times SO(1)$. The position vector b_i of part i is dependent on the augmented state vector r_a as follows: $b_i = r + d \begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix}$ where d denotes the mating distance between the robot and the mated part. Control laws are defined by again considering a collection of smooth maps of $\psi_i(r_a, \bar{b}_i) : \Re^2 \times SO(1) \times \Re^{2p-2} \rightarrow \Re$, $\forall i \in P$ as follows [2]:

$$\psi_i(r_a, \bar{b}_i) = \frac{\gamma_i^{k_3}(r_a, \bar{b}_i)}{\beta_i(r_a, \bar{b}_i)} \tag{2.4}$$

where $\bar{b}_i = \{b_1, \cdots, b_{i-1}, b_{i+1}, \cdots, b_p\}$. The parameter $k_3 \in \mathcal{Z}^+$ is chosen to be a positive integer. The function $\gamma_i(r_a, \bar{b}_i) : \Re^2 \times SO(1) \times \Re^{2(p-1)} \rightarrow \Re$ is the total squared Euclidean distance between the current configuration and the goal configuration:

$$\gamma_i(r_a, \bar{b}_i) = \left\| r + d \begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix} - g_i \right\|^2 + \sum_{j \in P}^{j \neq i} \|b_j - g_j\|^2 \tag{2.5}$$

The function $\beta_i(r_a, \bar{b}_i) : \Re^2 \times SO(1) \times \Re^{2(p-1)} \rightarrow \Re$, $\forall i \in P$, encodes the obstacles as:

$$\beta_i(r_a, \bar{b}_i) = \prod_{j \in P}^{j \neq i} \left[\|r - b_j\|^2 - (\rho_r - \rho_j)^2 \right] \times \left[\left\| r + d \begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix} - b_j \right\|^2 - (\rho_i - \rho_j)^2 \right] \tag{2.6}$$

The motion of robot carrying part i is defined by:

$$\dot{r}_a = -D_{r_a}\psi_i(r_a(t), \bar{b}_i(t)) \tag{2.7}$$

where $D_{r_a}\psi_i$ is the gradient of ψ_i with respect to r_a.

3. UNCERTAINTY REDUCTION BY PARTICLE AND KALMAN FILTERS

3.1. Robot Dynamics Under Noise

The robot dynamics as given in Chapter 2 are based on perfect sensory data. In case of noisy measurements, they cease to be valid and need to be modified in order to incorporate noise. First as the incoming sensory data is subjected to sampling, the continuous state function r becomes a discrete function r_k where the subscript $k \in \{0, 1, \cdots\}$ denotes discrete time index, which is $r(t)$ evaluated at time $t = k\Delta t$. Next, the presence of measurement and process noise turns the system into a stochastic process. As the system is subject to a process noise η_k, the system transition function f needs to be modified accordingly. Finally, as the state vector is subject to measurement noise ν_k, it is no longer possible to observe the state vector directly. Rather, one has the vector of observations z_k as defined by the measurement function h.

$$\begin{aligned} r_{k+1} &= f(r_k, \eta_k) \\ z_k &= h(r_k, \nu_k) \end{aligned} \tag{3.1}$$

The process noise η_k and measurement noise ν_k are assumed to be white and independent. Let us note that with PFs, the noise distributions need not to be known. All that is required is an assumption about the model of the underlying statistics – albeit with unknown parameters. In our case, they are both assumed to be Gaussian as $\eta \sim \mathcal{N}(\eta; 0, \Sigma_\eta)$ and $\nu \sim \mathcal{N}(\nu; 0, \Sigma_\nu)$, respectively where the covariances Σ_η and Σ_ν are assumed to be known. This seems a valid model for the visual feedback used by our robot. Furthermore, with this assumption and our nonlinear state space model, optimal importance function as proposed in [8] can be used – which improves the performance of the PF.

3.1.1. Robot Motion State and Observation Model

Under noisy measurements, the state dynamics for the *mate-part* now incorporate dynamical noise as:

$$\dot{r}(t) = -D_r \varphi_i(r(t), b(t)) + \eta(t) \tag{3.2}$$

In the *move-part* stage, similarly the system dynamics should now incorporate noise as well:

$$\dot{r}(t) = -D_r \psi_i(r_a(t), \bar{b}_i(t)) + \eta(t) \tag{3.3}$$

Due to measurement noise, the robot state $r(t)$ is no longer available. Rather, we have its measurement $z(t)$.

$$z(t) = r(t) + \nu(t) \tag{3.4}$$

3.1.2. Part State and Observation Model

The state equations governing the parts' motion are linear because parts are stationary unless the robot carries them. The system model is as follows:

$$\dot{\bar{b}}_i(t) = \eta(t) \tag{3.5}$$

Similarly, we have only observations $z_{\bar{b}_i}$ of $\bar{b}_i$

$$z_{\bar{b}_i}(t) = \bar{b}_i(t) + \nu(t) \tag{3.6}$$

3.2. Introducing Particle Filters

PFs provide an estimate $\hat{r}_k$ of r_k. This is accomplished by weighted approximation of the posterior density function $p(r_k|z_k)$ from samples z_k of $z(t)$ in a recursive manner. The approximation equations are derived considering a very general, nonlinear and non-Gaussian system. Consequently, no assumptions such as the linearity and Gaussian noise as required by the classical Kalman filter need to be made. The capability to handle nonlinear, non-Gaussian systems allows PFs to achieve improved accuracy over Kalman filter-based estimation methods. The more nonlinear the model is, or the more non-Gaussian the noise is, the more potential the PF has, especially in applications where computational power is rather cheap and the sampling rate is slow [19].

3.2.1. Markov Chain

The system dynamics can be interpreted as an hidden Markov model. Hence, it is defined by three pdfs:

- $p(r_0)$: The initial pdf of the random variable r_k
- $p(r_k|r_{k-1})$: The likelihood of r_k given r_{k-1}, r_k evolve according to a Markov chain with transition probabilities given by $p(r_k|r_{k-1})$.
- $p(z_k|r_k)$: The likelihood of z_k given r_k.

In this framework, the state r_k needs to estimated given the history of observations (measurements) $z_{0:k} = z_0, ..., z_k$ up to time k. The solution to this problem is given by the posterior density $p(r_k|z_{0:k})$, or in other words, the likelihood of the current state given the history of observations. PFs approximate the *posterior* pdf under nonlinear and nongaussian system dynamics of the Markov chain based on a finite number of samples with associated weights [9].

3.2.2. Recursive State Estimation

The observations become available one at a time, which then motivates a recursive framework to allow online estimation. The posterior probability density can be recursively updated using the following equations [17]:

$$p(r_k|z_{0:k}) = \frac{p(z_k|r_k)p(r_k|z_{0:k-1})}{p(z_k|z_{0:k-1})} \tag{3.7}$$

$$p(r_k|z_{0:k-1}) = \int p(r_k|r_{k-1})p(r_{k-1}|z_{0:k-1})dr_k \tag{3.8}$$

$$p(z_k|z_{0:k-1}) = \int p(z_k|r_k)p(r_k|z_{0:k-1})dr_k \tag{3.9}$$

The conditional pdfs $p(r_k|r_{k-1})$ and $p(z_k|r_k)$ are known from model (3.1). This scheme consists of a measurement update according to (3.7) and (3.9) along with time update according to (3.8) as derived in Appendix A. These equations look deceptively simple. However, they are really theoretical as the (multidimensional) integrals involved rarely yield an analytical solution. The linear, Gaussian case is an exception and its solution is known as the Kalman filter. PFs numerically approximate the posterior pdf $p(r_{0:k}|z_{0:k})$ using MC techniques. The approximation of $p(r_k|z_{0:k})$ is obtained by marginalisation.

3.2.3. Monte Carlo Integration

MC integration is based on the assumption that it is possible to draw $N >> 1$ samples $r^i, i = 1, ..., N$ distributed according to $\pi(r)$. Accordingly, the multidimensional integral

$$I = \int f(r)\pi(r)dr \tag{3.10}$$

can be approximated by the sample mean:

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(r^i). \tag{3.11}$$

In the Bayesian estimation context, the function $\pi(r)$ is the posterior density. Samples from $p(r_{0:k}|z_{0:k})$ have to be drawn, but that density is only known up to a proportionality constant. This problem can be circumvented using importance sampling.

3.2.4. Sequential Importance Sampling

Ideally samples are generated directly from $\pi(r)$ and I can be estimated using Equation 3.11. Suppose samples can only be generated from a density $q(r)$ which is close to, but not exactly equal to $\pi(r)$. Then a correct weighting of the sample set still makes the MC estimation possible. The pdf $q(r)$ is referred to as the *importance* or *proposal* density. Importance sampling is a modification of perfect sampling. The samples are drawn from the importance function (3.10) can be rewritten as:

$$I = \int f(r)\pi(r)dr = \int f(r)\frac{\pi(r)}{q(r)}q(r)dr \tag{3.12}$$

A MC estimate of I is computed by generating $N >> 1$ independent samples $r^i, i =, ..., N$ distributed according to $q(r)$ and forming the weighted sum:

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(r^i)\tilde{w}(r^i) \tag{3.13}$$

where $\tilde{w}(r^i) = \frac{\pi(r^i)}{q(r^i)}$ are the importance weights. The choice of the importance or proposal density (function) is critical with respect to performance. The optimal importance function that minimizes the variance of the importance weights, conditioned on $r_{0:k-1}$ and $z_{0:k}$ has been shown to be [8]:

$$q(r_k^i|r_{0:k-1}^i, z_{0:k})_{opt} \quad = \quad p(r_k^i|r_{k-1}^i, z_k)$$

$$= \frac{p(z_k|r_k, r_{k-1}^i)p(r_k|r_{k-1}^i)}{p(z_k|r_{k-1}^i)} \tag{3.14}$$

The SIS algorithm is a MC method that forms the basis for most SMC filters developed over the past decades. This SMC approach is known variously as bootstrap filtering [20], the condensation algorithm [21], and particle filtering [22].

If the normalizing factor of the desired density $\pi(r)$ is unknown, the importance weights are needed to be normalized. Then I_N is as follows:

$$I_N = \frac{\frac{1}{N}\sum_{i=1}^{N} f(r^i)\tilde{w}(r^i)}{\frac{1}{N}\sum_{j=1}^{N} \tilde{w}(r^j)} = \sum_{i=1}^{N} f(r^i)w(r^i) \tag{3.15}$$

where the normalized importance weights are given by $w(r^i) = \tilde{w}(r^i)/\sum_{j=1}^{N} \tilde{w}(r^j)$.

3.3. Algorithm

The algorithm consists of two stages: prediction and updating. In prediction, the estimated value $\hat{r}_k$ is predicted from the previous observations. Following, the posterior probability $p(r_k|z_k)$ is updated using the newly acquired measurements. It is represented using a set consisting of N particles $\{r_k^i \mid i = 1, \cdots, N\}$ with associated weights w_k^i, for each time step k as:

$$p(r_k|z_k) \approx \sum_{i=1}^{N} w_k^i \delta(r_k - r_k^i) \tag{3.16}$$

It can be shown that as $N \rightarrow \infty$, the approximation approaches the true posterior[9]. The weights are updated within a normalization using the importance sampling as :

$$w_k^i \propto w_{k-1}^i \frac{p(z_k|r_k^i)p(r_k^i|r_{k-1}^i)}{q(r_k^i|r_{0:k-1}^i, z_{0:k})} \tag{3.17}$$

where $r_{0:k-1} = [r_0, r_1, .., r_{k-1}]^T$ and $z_{0:k} = [z_0, z_1, .., z_k]^T$. Note that the state equation characterizes the state transition probability $p(r_k|r_{k-1})$, whereas the measurement

equation describes the likelihood $p(z_k|r_k)$.

The substitution of (3.14) into (3.17) yields

$$w_k^i \propto w_{k-1}^i p(z_k|r_{i-1}^i) \tag{3.18}$$

It requires sampling from $p(r_k|r_{k-1}^i, z_k)$ and the evaluation of $p(z_k|r_{k-1}^i)$, which are both realizable in our case. Under our assumptions on the state and observation models, both the optimal importance density and $p(z_k|r_{k-1})$ are Gaussian [8]:

$$p(r_k^i|r_{k-1}^i, z_k) = \mathcal{N}(r_k^i; m_k, \Sigma_k) \tag{3.19}$$

$$p(z_k|r_{k-1}^i) = \mathcal{N}(z_k; f_{k-1}, \Sigma_\eta + \Sigma_\nu) \tag{3.20}$$

where

$$\Sigma_k^{-1} = \Sigma_\eta^{-1} + \Sigma_\nu^{-1} \tag{3.21}$$

$$m_k = \Sigma_k \left(\Sigma_\eta^{-1} f_{k-1} + \Sigma_\nu^{-1} z_k\right) \tag{3.22}$$

$$f_k = \begin{cases} -\int D_r \varphi_i(r_k, b_k) & \text{if } state = mate - part \\ -\int D_{r_a} \psi_i(r_{a_k}, \bar{b}_{i_k}) & \text{if } state = move - part \end{cases} \tag{3.23}$$

where $\mathcal{N}(r; \mu, \Sigma)$ is a Gaussian density with argument r, mean μ, and covariance Σ.

The weights at time k can be computed and normalized such that $\sum_i w_k^i = 1$ before the particles are propagated to time k. One can perform resampling, if necessary, to obtain an approximate i.i.d. sample from $p(r_k|z_k)$.

This algorithm forms the basis for the PF which consists of the recursive evolution of the importance weights and the particles as measurements are received sequentially. A pseudo-code description of the algorithm is given in Table 3.1.

Without resampling in step-5 of Table 3.1, during the computation of the weights, most particles may converge to 0 which is a common problem known as the degener-

Table 3.1. Pseudocode for SIR particle filter

1. Initialization

   For i = 1:N

   - Draw $r_k^i \sim \mathcal{N}(r_k^i; z_0, \Sigma_\nu)$

   End For

2. Sample from the optimal importance density

   For i = 1:N

   - Draw $r_k^i \sim \mathcal{N}(r_k^i; m_k, \Sigma_k)$
   - Assign the particle a weight, $\tilde{w}_k^i$, according to eq. (3.18)

   End For

3. Compute normalized importance weights

   For i = 1:N

   - $w_k^i = \tilde{w}_k^i / SUM\left[\{\tilde{w}_k^i\}_{i=1}^N\right]$

   End For

4. Calculate N_{eff} according to eq. (3.24)
5. IF $N_{eff} < N_{kh}$

   Resample using systematic resampling

   $\{r_k^i, w_k^i\}_{i=1}^N = RESAMPLE\left[\{r_k^i, w_k^i\}_{i=1}^N\right]$
6. Estimate $\hat{r}_k$, weighted sum of particles
7. Simulate the system using estimated $\hat{r}_k$
8. Go to step-2

acy phenomenon [8]. Hence, the filter would not estimate $\hat{r}_k$ from noisy observations properly. The particles are resampled whenever a significant degeneracy is observed, that is, whenever N_{eff} falls below a certain threshold N_{th}. The threshold was chosen as $N_{th} = 2N/3$. The effective sample size N_{eff} introduced in [23] is estimated as:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} \left(w_k^i\right)^2} \tag{3.24}$$

In the resampling stage, N particles are taken with replacemant from $\{r_k^i\}_{i=1}^N$, where the probability to take particle i is w_k^i. Systematic resampling [9] is used in step-5 of Table 3.1. Appendix B shows details of the sampling algorithm.

3.4. Kalman Filter

The Kalman filter is used to provide an estimate $\hat{b}_k$ of b_k, which is $b(t)$ discretized at time $t = k\Delta t$. Like the PF, there are two stages in the estimation procedure-the prediction and the update. However, the Kalman filter assumes, at every time step k, that the posterior density is Gaussian and hence characterized by a mean and a covariance. It can be proved that if $p(b_{k-1}|z_{k-1})$ is Gaussian then $p(b_k|z_k)$ is also Gaussian provided that certain assumptions hold: The process and the measurement noise are both Gaussian distributions of known parameters and both the state equation and the measurement equation are known linear functions of the relevant states and noise [24]. These assumptions are all satisfied for the parts' state and observation models in our case. With the assumption of a prior estimate $\hat{b}_k^-$ and a prior error covariance matrix P^-, the measurement z_b is used to improve the prior estimate as in step-3 of Table 3.2. The updated estimated $\hat{b}_k$ is easily projected ahead via the transition matrix as in step-5 of Table 3.2. K is called the gain matrix, and P is called the covariance matrix of the prediction error.

The Kalman filter algorithm, which is derived using (3.7) and (3.9), can be sum-

Table 3.2. Pseudocode for Kalman filter

1. Prior estimate $\hat{b}_k^-$, error covariance matrix P_k^-
2. Compute Kalman gain
   $K_k = P_k^- \left(P_k^- + \Sigma_\nu\right)^{-1}$
3. Update estimate with measurement z_b
   $\hat{b}_k = \hat{b}_k^- + K_k \left(z_{b,k} - \hat{b}_k^-\right)$
4. Compute error covariance matrix for updated estimate
   $P_k = (I - K_k) P_k^-$
5. Project ahead
   $\hat{b}_{k+1}^- = \hat{b}_k$
   $P_{k+1}^- = P_k + \Sigma_\eta$
6. Go to step-2

marized as follows [9]:

$$p(b_{k-1}|z_{b,1:k-1}) = \mathcal{N}(b_{k-1}; \hat{b}_k, P_{k-1}) \tag{3.25}$$

$$p(b_k|z_{b,1:k-1}) = \mathcal{N}(b_k; \hat{b}_k^-, P_k^-) \tag{3.26}$$

$$p(b_k|z_{b,1:k}) = \mathcal{N}(b_k; \hat{b}_{k+1}, P_k) \tag{3.27}$$

where

$$\hat{b}_{k+1}^- = \hat{b}_k \tag{3.28}$$

$$P_{k+1}^- = P_k + \Sigma_\eta \tag{3.29}$$

$$\hat{b}_k = \hat{b}_k^- + K_k \left(z_{b,k} - \hat{b}_k^-\right) \tag{3.30}$$

$$P_k = (I - K_k) P_k^- \tag{3.31}$$

and where $\mathcal{N}(r; \mu, \Sigma)$ is a Gaussian density with argument r, mean μ, and covariance Σ.

3.5. Cramer-Rao Lower Bound

The PF and Kalman filter provide an estimate of both the robot's and parts' positions, respectively. Next, the theoretical performance of this estimator is assessed given a model structure. In particular, this analysis is done using the Fisher information matrix (FIM), which gives a bound on the second order moment. This bound is often referred to as the Cramer-Rao lower bound (CRLB) [10, 25].

In nonlinear estimation, Root Mean Square (RMS) error may be based on the region of the state dynamics in which the simulation is carried out. This is for example the case in the parts' moving problem, where the simulations in *move-part* are more susceptible to noise than in *mate-part*. Hence, the CRLB yields an opportunity to evaluate RMS performace of filters under different circumstances [26].

Let $\hat{r}$ be an unbiased estimator of the state vector r, based on the measurement z and prior knowledge of initial density $p(r_0)$. Then the CRLB for the error covariance matrix is defined to be the inverse of the Fisher information matrix $\mathbf{J}$ [27]:

$$\mathbf{P} \triangleq \mathbb{E}\left\{[\hat{r} - r]\,[\hat{r} - r]^T\right\} \geq \mathbf{J}^{-1} \tag{3.32}$$

where $\mathbf{J}$ is the 2x2 Fisher information matrix with the elements

$$\mathbf{J}_{ij} = \mathbb{E}\left[-\frac{\partial^2 logp(z|r)}{\partial r[i]\partial r[j]}\right] i, j = 1, 2 \tag{3.33}$$

provided that the derivatives and expectations exist. The inequality means that the difference $\mathbf{P} - \mathbf{J}^{-1}$ is a positive semi-definite matrix.

Since the estimates $\hat{r}_k$ are obtained iteratively in PF, the posterior CRLB is defined as:

$$\mathbf{P}_k \triangleq \mathbb{E}\left\{[\hat{r}_k - r_k]\,[\hat{r}_k - r_k]^T\right\} \geq \mathbf{J}_k^{-1} \tag{3.34}$$

Increasing discrete time index k results in the increasing demand for both computational and memory resources. However, the CRLB can be computed recursively. Tichavsky et al. [28] provide a method of computing the information matrix $\mathbf{J}_k$ recursively for Gaussian noise case:

$$\mathbf{J}_{k+1} = \mathbf{K}_k^{22} - \mathbf{K}_k^{21} \left(\mathbf{J}_k + \mathbf{K}_k^{11}\right)^{-1} \mathbf{K}_k^{12} \tag{3.35}$$

where

$$\begin{aligned} \mathbf{K}_k^{11} &= \mathbb{E}\left\{\tilde{\mathbf{F}}_k^T \Sigma_\eta^{-1} \tilde{\mathbf{F}}_k\right\} \\ \mathbf{K}_k^{12} &= -\mathbb{E}\left\{\tilde{\mathbf{F}}_k^T\right\} \Sigma_\eta^{-1} = \left(\mathbf{K}_k^{21}\right)^T \\ \mathbf{K}_k^{22} &= \Sigma_\eta^{-1} + \mathbb{E}\left\{\tilde{\mathbf{H}}_k^T \Sigma_\nu^{-1} \tilde{\mathbf{H}}_k\right\} \end{aligned}$$

Here, f is the system transition function and h is the measurement function. $\tilde{\mathbf{F}}_k = D_r\, f|_{r_k}$ and $\tilde{\mathbf{H}}_k = D_r\, h|_{r_k}$ are Jacobians of f and h evaluated at the true value of r_k, respectively.

The discrete-time version of dynamics is obtained using the Euler approximation with integration step τ, that is:

$$r_{k+1} = f(r_k) + \eta_k \tag{3.36}$$

where

$$f(r_k) = \left\{ \begin{array}{ll} r_k - \tau D_r \varphi_i(r_k, b_k) & \text{if } state = mate - part \\ r_k - \tau D_r \psi_i(r_{a_k}, \bar{b}_{i_k}) & \text{if } state = move - part \end{array} \right\} \tag{3.37}$$

The analysis of CRLB can be restricted to the zero process noise case, since it is observed that small amounts of process noise have only negligible effect. In the absence of process noise, $\Sigma_\eta = 0$, the corresponding term vanishes and the information matrix

J simplifies to:

$$\mathbf{J}_{k+1} = \left(\tilde{\mathbf{F}}_k^{-1}\right)^T \mathbf{J}_k \tilde{\mathbf{F}}_k^{-1} + \Sigma_\nu^{-1} \tag{3.38}$$

where $\tilde{\mathbf{F}}_k$ is the Jacobian of f evaluated at the true state r_k; that is,

$$\tilde{\mathbf{F}}_k = \bigtriangledown_{r_k} f|_{r_k} \tag{3.39}$$

In the case where the initial distribution is Gaussian, that is, $p(r_0) = \mathcal{N}(r_0; 0, \Sigma_\nu)$, the initial information matrix $\mathbf{J}_0$ is calculated from the initial density $p(r_0)$ [10]:

$$\mathbf{J}_0 = \Sigma_\nu^{-1} \tag{3.40}$$

The CRLB of the components of r_k are calculated as the diagonal elements of the inverse of the information matrix $\mathbf{J}_k$. If $\mathbf{J}_k^{-1}[i,j]$ denotes the ijth element of the inverse information matrix, then the corresponding CRLB can be written as

$$CRLB(RMS_k) = \sqrt{\mathbf{J}_k^{-1}[1,1] + \mathbf{J}_k^{-1}[2,2]} \tag{3.41}$$

4. SIMULATIONS

A series of simulations have been conducted for three and six parts scenarios of varying degrees of task complexity and sensor noise. Task complexity is measured by how tightly packed the parts need to be at their final positions. It is defined by $comp = (100\binom{p}{2}/\log\beta)$ where the scalar function $\beta = \prod_{(i,j)\in P}\left[\|g_i - g_j\|^2 - (\rho_i - \rho_j)^2\right]$ [2]. Depending on its value, the task complexity is varied from easy (E) to intermediate (I) and to finally hard (H).

Similarly, the measurement noise level is selected as low noise (variance = 0.2), medium noise (variance = 1.0) and high noise (variance = 5.0) according to signal to noise ratio (SNR) defined as:

$$SNR = 10\ log(\frac{e}{\sigma_\nu^2}) \tag{4.1}$$

$$e = \frac{1}{T}\int_0^T V^2 dt \tag{4.2}$$

where e is the signal power, T is execution time for simulation, V is velocity of the robot, and σ_ν^2 is variance of the measurement noise. The process noise variance is 0.06.

In order to evaluate performance, a series of performance measures similar to those presented in [2] have been used:

- Normalized part length (npl):

$$npl = \frac{1}{p}\sum_{i\in P}\frac{\int_0^{t_f}\left\|\dot{b}_i(t)\right\|dt}{\|b_i(0) - g_i\|} \tag{4.3}$$

where t_f denotes the duration of a task.

- Normalized robot part length (nrl):

$$nrl = \frac{\int_0^{t_f} \|\dot{r}_i(t)\| \, dt}{\sum_{i \in P} \|r(0) - b_i(0)\| - (\rho_i + \rho_r) + \sum_{(i,j) \in P} \|g_i - g_j\|} \tag{4.4}$$

  where $r(0)$ denotes the initial position of the robot.

- Positional inaccuracy (pi):

$$pi = \frac{1}{p} \sum_{i \in P} \frac{1}{\rho_i} \|b_i(t_f) - g_i\| \tag{4.5}$$

- Robot positional estimation error ($rpee$):

$$rpee = \frac{1}{t_f} \sum_{t=0}^{t_f} \|r(t) - \hat{r}_t\| \tag{4.6}$$

- Part positional estimation error ($ppee$):

$$ppee = \frac{1}{t_f} \sum_{t=0}^{t_f} \left\| b(t) - \hat{b}_t \right\| \tag{4.7}$$

However, differing from previous definitions, $nnrl$, $nnpl$ and pi measures are calculated by normalizing nrl, npl and pi values by nrl, npl and pi in no noise case, respectively. Hence, their closeness to 1 indicate similarity of performance to ideal conditions.

The simulations are first performed without any state estimation under different noise levels. Each experiment is repeated with 10 random initial configurations. Next, same experiments are repeated with 49, 100 and 225 particles. In the figures, each data point represents the mean of 10 sample run with random initial configuration. The posterior pdf is supergaussian as associated kurtosis is calculated at each iteration and found to be in the range of 1.5 to 2. This verifies that conventional Kalman filtering cannot be utilized in our nonlinear robot system. The classical measure of

nongaussianity is the kurtosis or the fourth order cumulant and is defined by:

$$K(r) = \frac{E[(r-\mu)^4]}{\sigma^4} \tag{4.8}$$

where r is the random variable, μ is mean (first moment) of the distribution and σ^4 is fourth moment of the distribution.

4.1. 3-Part Simulations

The goal configurations for the 3-part simulations are shown in Figure 4.1.

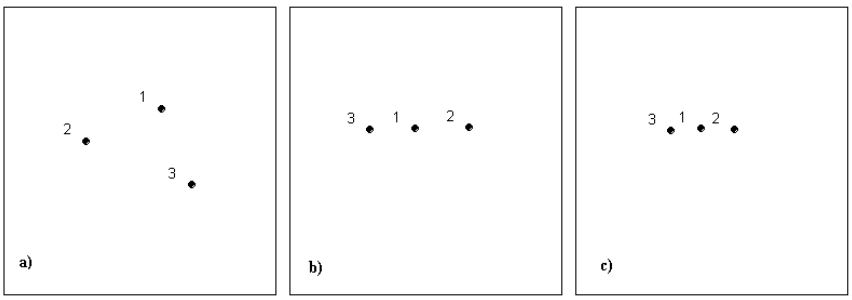

Figure 4.1. Goal configurations for 3-part case with increasing task complexity: a) Easy, b) Intermediate, c) Hard

4.1.1. Performance Measures for 3 Parts

The performance measures are as shown in Figure 4.2, 4.3 and 4.4. For low noise, there is only slight improvement in performance by using PF. Performance does not vary much with added task complexity. Furthermore, increasing the number of particles does not change the performance – in particular with respect to estimation errors in the robot's and parts' positions as the noise is not high enough to deteriorate the performance. In medium and high noise, a notable decrease in variance of *rpee* occurs, which results in lowered *nrl*, *npl* and *pi* values - particular for the case of 225 particles. This is expected since there is a considerable improvement in the estimate of the instantaneous state information with the particle and Kalman filters. The positional error in the parts' position is nearly the same in all task difficulty levels and is independent from the number of particles because Kalman filtering is applied to estimate

parts' position.

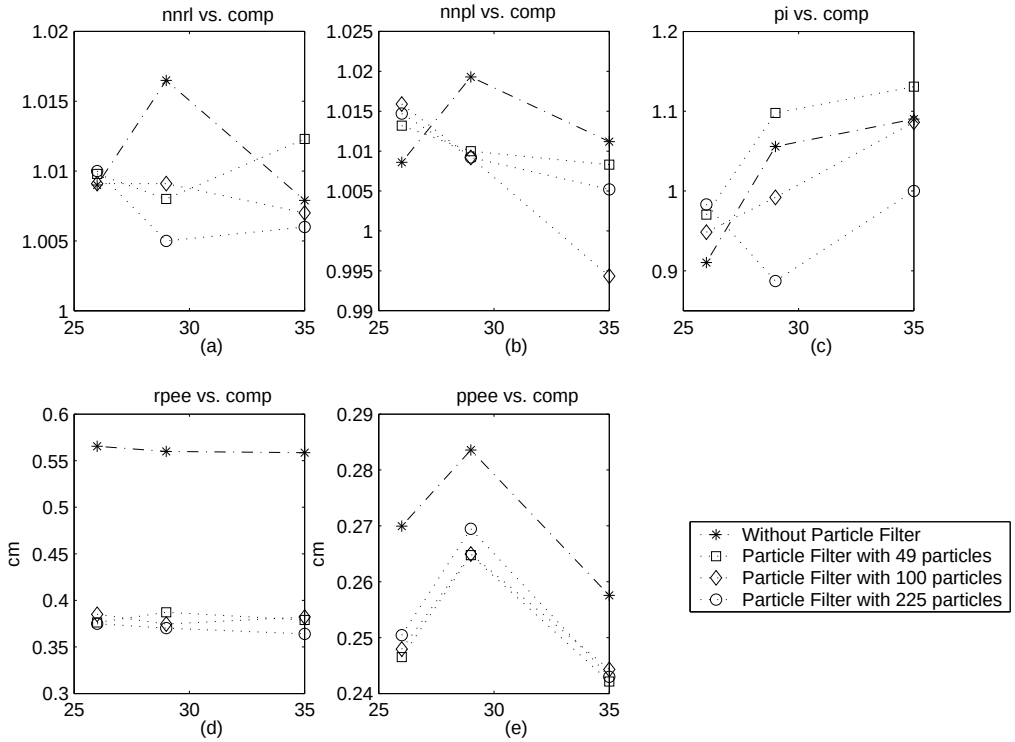

Figure 4.2. Performance measures vs. *comp* in 3-part and low noise case

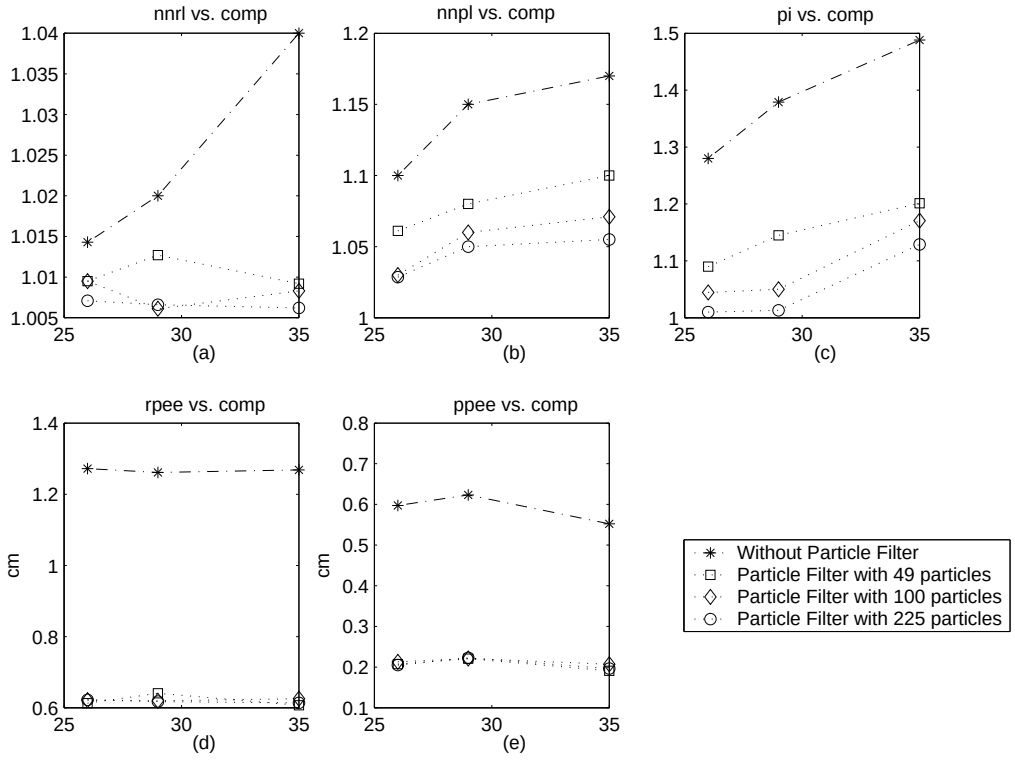

Figure 4.3. Performance measures vs. *comp* in 3-part and medium noise case

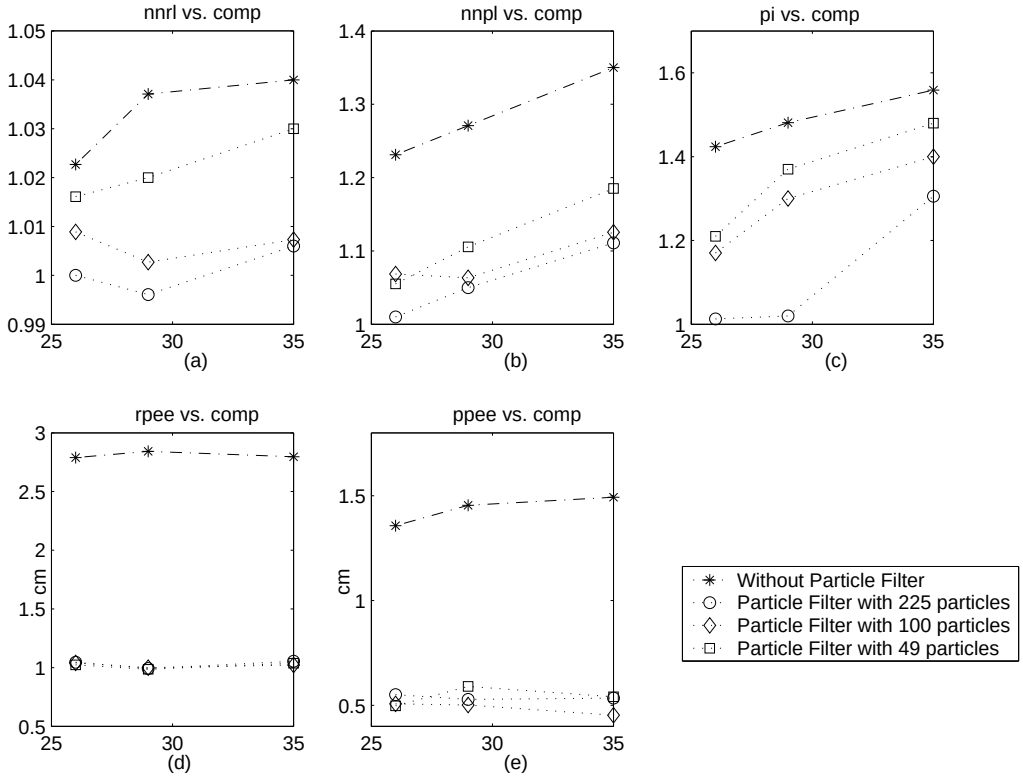

Figure 4.4. Performance measures vs. *comp* in 3-part and high noise case

The collision percentages for each task are shown in Table 4.1. As expected, the collision percentages increase as a function of task complexity and noise level. Furthermore, increased number of particles decrease the number of collisions considerably.

Table 4.1. Collision percentages in 3-part tasks

	Low Noise			Medium Noise			High Noise		
Num. of Particle	E	I	H	E	I	H	E	I	H
0 (without PF)	0	10	10	0	30	30	10	40	50
49	0	0	0	0	10	10	0	10	20
100	0	0	0	0	0	10	0	10	20
225	0	0	0	0	0	0	0	0	10

4.1.2. Cramer-Rao Error Performance for 3 Parts

Cramer-Rao bound is computed for the 3 parts easy, intermediate and hard scenarios with 100 particles. In the figures, each data point represents the mean of 20 sample runs. Over the course of the task completion, as the robot switches among the different mating and moving subtasks, the computed bounds are as shown in Figure 4.5, 4.6 and 4.7. In case of low and medium noises, the bound is almost constant. But, in high noise, the bound fluctuates considerably. It is observed that with increased noise levels, the effect of PFs on reducing the error becomes more pronounced. It is further noted that the bound level in *move-part* stage is more rough than in *mate-part* stage. This is because the system dynamics in *move-part* stage is more complex than in *mate-part* stage.

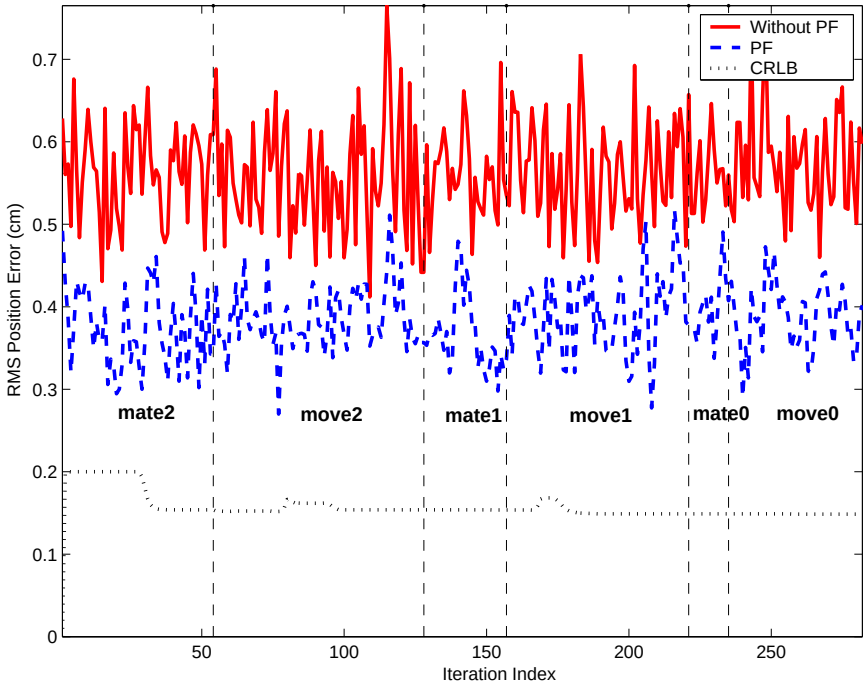

Figure 4.5. The performance of PF vs. CRLB in 3-part and low noise

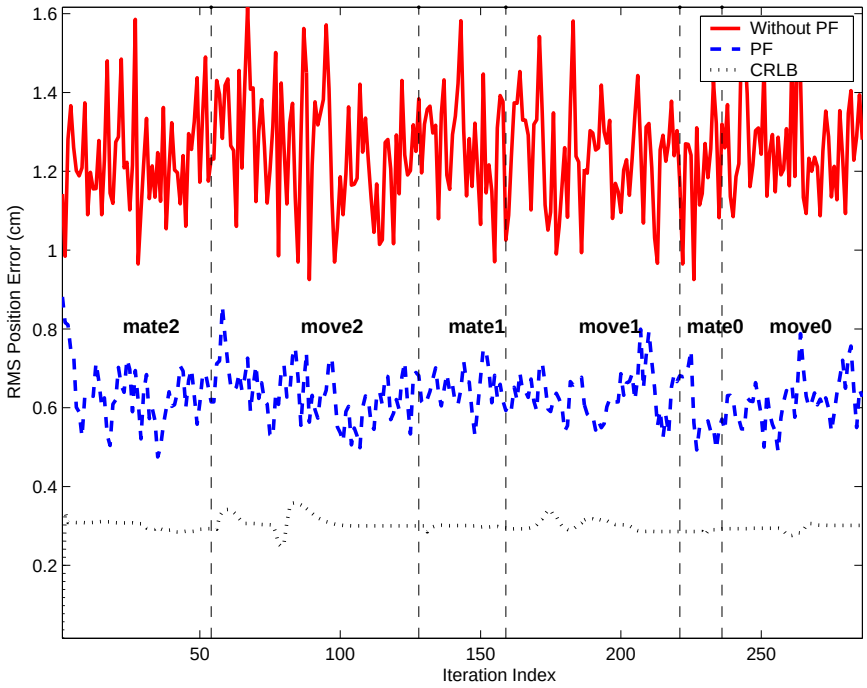

Figure 4.6. The performance of PF vs. CRLB in 3-part and medium noise

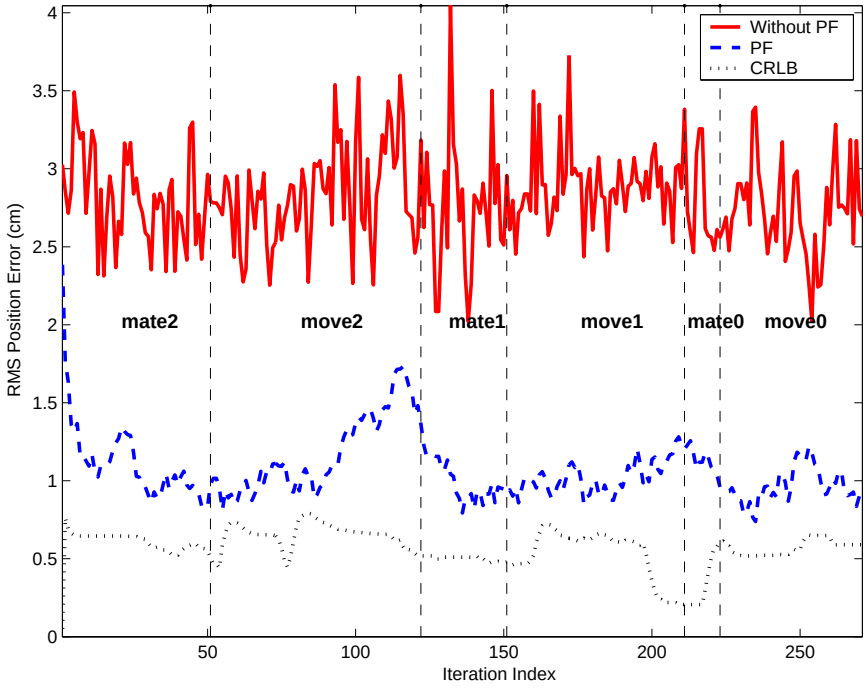

Figure 4.7. The performance of PF vs. CRLB in 3-part and high noise

4.2. 6-Part Simulations

The goal configurations for the 6-part simulations are shown in Figure 4.8.

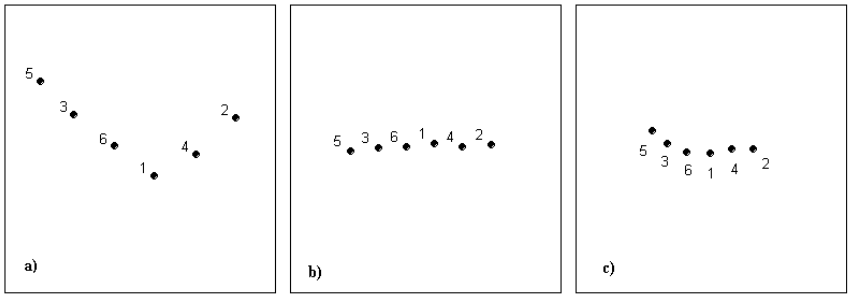

Figure 4.8. Goal configurations of 6-part case with increasing task complexity: a) Easy, b) Intermediate, c) Hard

4.2.1. Performance Measures for 6 Parts

The performance results are presented in Figure 4.9, 4.10 and 4.11. These graphs reveal conclusions similar to those of the 3-part case.

The collision percentages for each task are shown in Table 4.2. Again, as expected, the collision percentages increase as a function of task complexity and noise level. For low and medium noise cases, there is a notable decrease in the collision percentages, especially using 100 and 225 particles in the scenario where the task difficulties are intermediate and hard. In high noise, the effect of PF and Kalman filter can be observed clearly. Performance measures *nrl*, *npl* and *rpee* in the 6-part case are close to in the 3-part case. However, the improvement in the collision percentages is better than in the 3-part case. This is expected since while increasing the number of parts, possible collisions are increasing without PF.

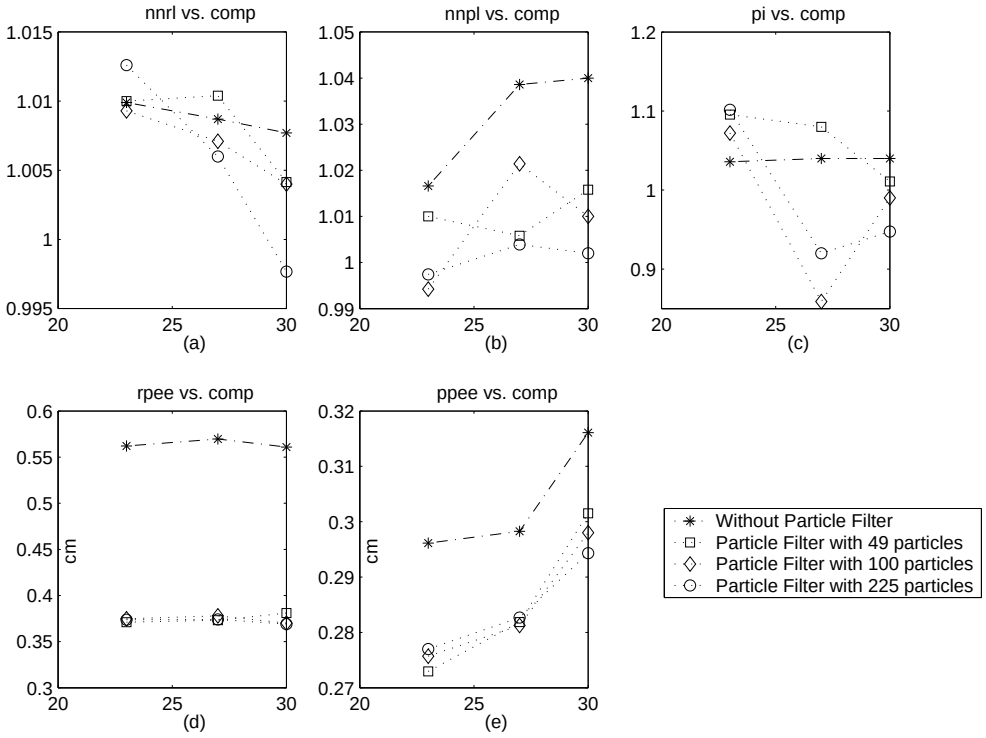

Figure 4.9. Performance measures vs. *comp* in 6-part and low noise case

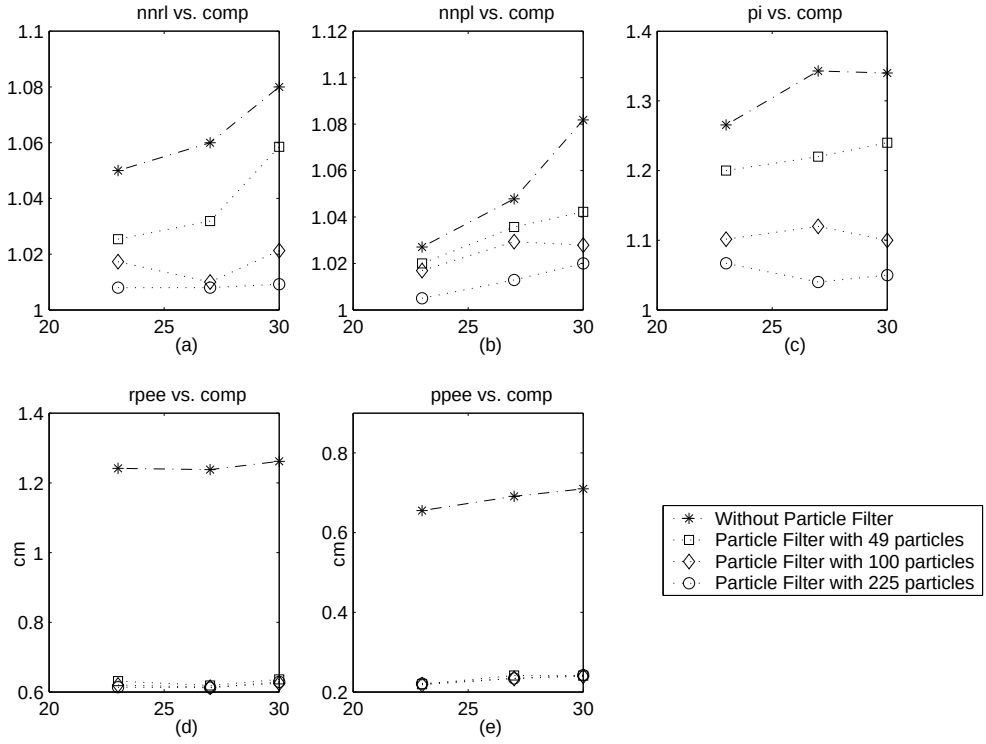

Figure 4.10. Performance measures vs. *comp* in 6-part and medium noise case

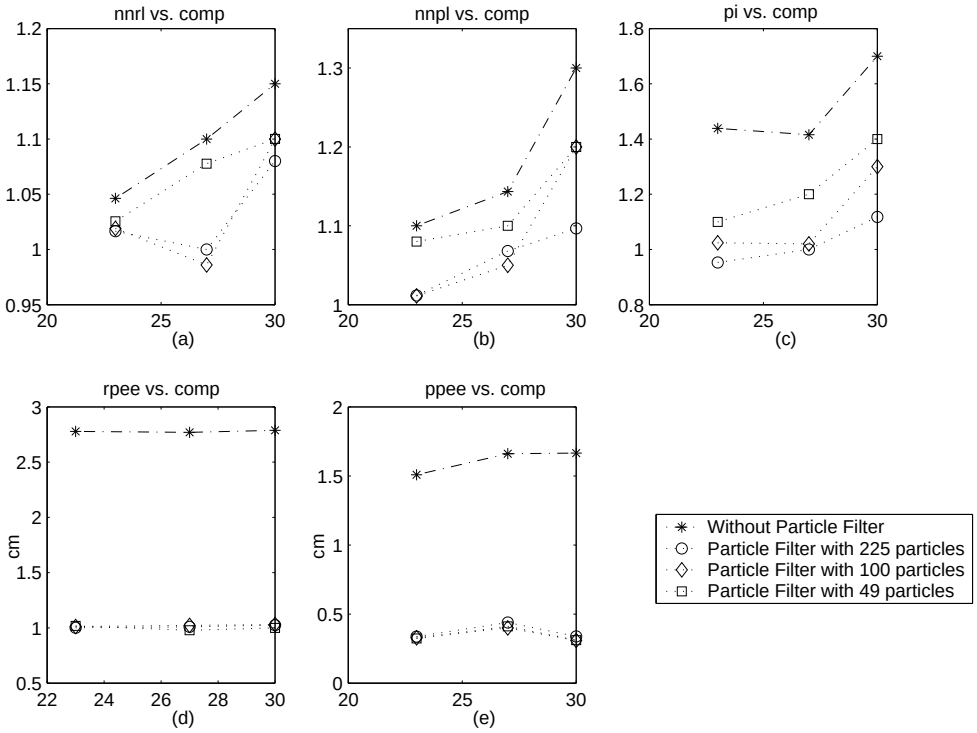

Figure 4.11. Performance measures vs. *comp* in 6-part and high noise case

Table 4.2. Collision percentages in 6-part tasks

	Low Noise			Medium Noise			High Noise		
Num. of Particle	E	I	H	E	I	H	E	I	H
0 (without PF)	0	30	50	0	50	60	10	70	70
49	0	10	30	0	10	30	0	20	40
100	0	10	20	0	10	20	0	20	30
225	0	0	10	0	0	10	0	10	20

4.2.2. Cramer-Rao Error Performance for 6 Parts

Cramer-Rao bound is computed for the 6 parts easy, intermediate and hard scenarios with 100 particles. Over the course of the task completion, as the robot switches among the different mating and moving subtasks, the computed bounds are as shown

in Figure 4.12, Figure 4.13 and Figure 4.14.

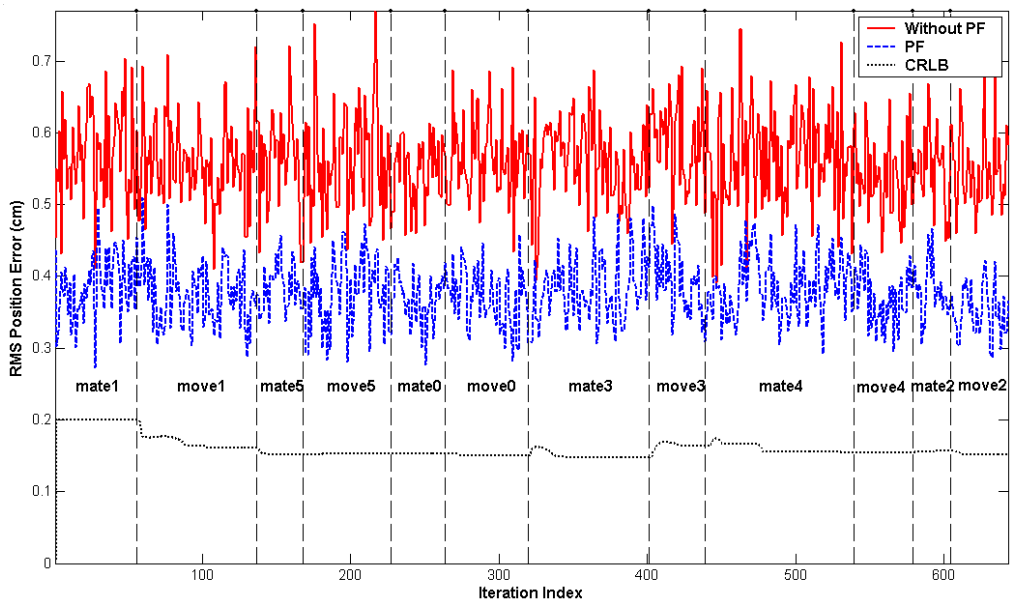

Figure 4.12. The performance of PF vs. CRLB in 6-part and low noise

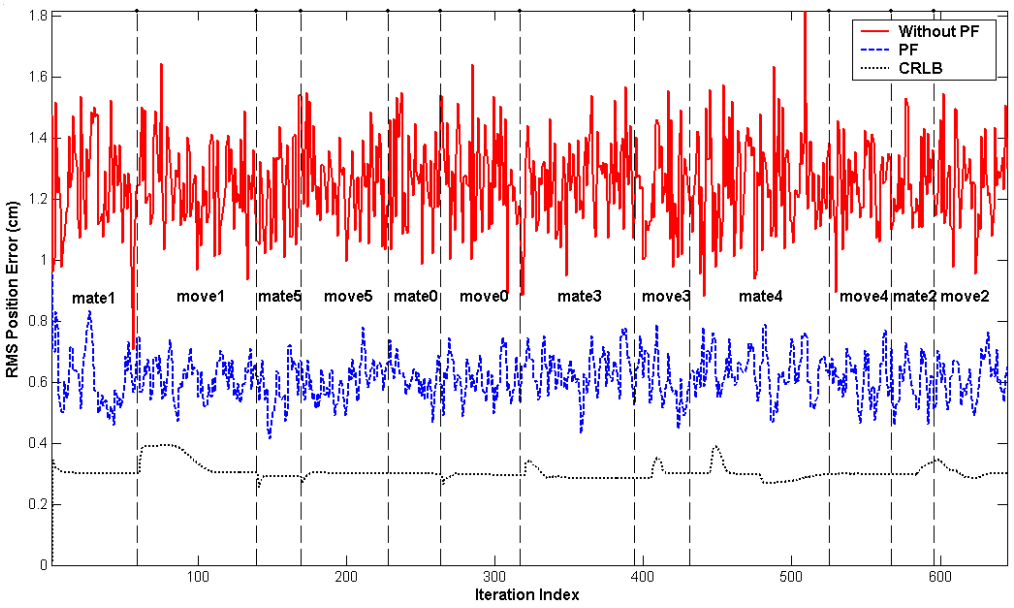

Figure 4.13. The performance of PF vs. CRLB in 6-part and medium noise

In case of low and medium noises, the bound is almost constant. But, in high noise, the bound fluctuates more than that in the 3-part case. However, the perfor-

mance of the PF is same as the 3-part case. It is observed that with increased noise levels, the effect of PFs on reducing the error becomes more pronounced.

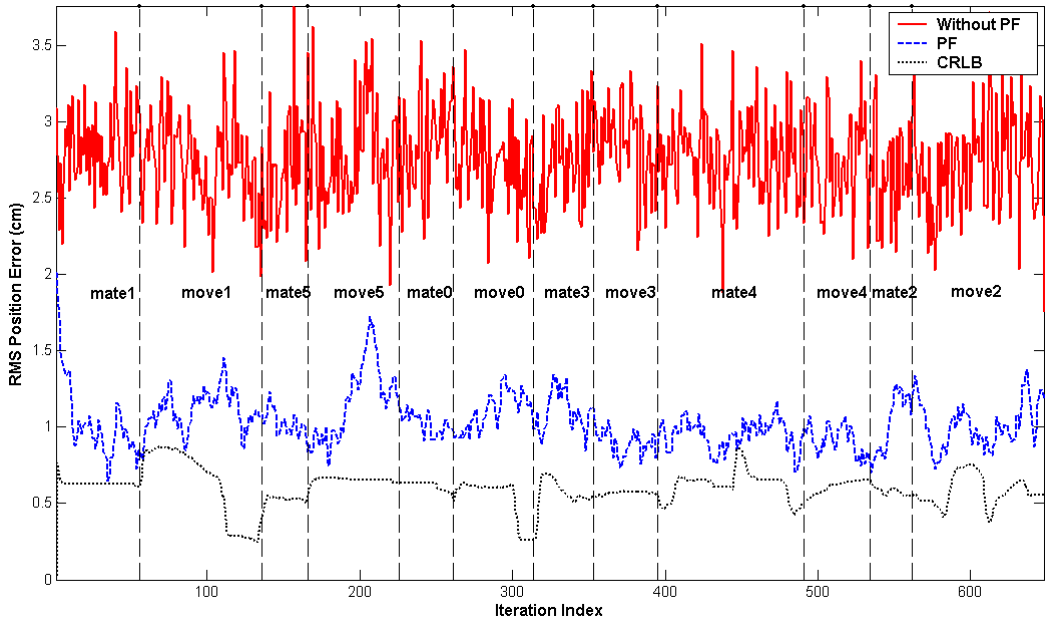

Figure 4.14. The performance of PF vs. CRLB in 6-part and high noise

4.3. Discussion

The PF runs on the desktop computer with AMD Athlon 2000+ 1.67 GHz processor and 1.0 GB RAM under Windows XP Professional operating system. The processing time of the PF in both *mate-part* and *move-part* stages is shown in Table 4.3. It is observed that the processing time of the PF in *move-part* stage is longer than in *mate-part* stage since the system dynamics in *move-part* stage is more complex than in *mate-part* stage.

Table 4.3. Processing time of PF (ms)

Num. of	Stages	
Particle	*mate-part*	*move-part*
49	40	90
100	72	173
225	150	380

The processing time of PF with 225 particles in *move-part* stage takes 380 milliseconds. This decreases the performance of the PF in real-time. Furthermore, although the processing time of the PF with 225 particles takes two times more than that of the PF with 100 particles, the performance does not increase by same ratio. Hence, the PF with 100 particles was used in experiments.

5. EXPERIMENTS

This chapter presents experimental results from an actual parts' rearrangement scenario. These experiments are conducted with EDAR - Event Driven Assembler Robot which is mobile robot with manipulation capabilities as shown in Figure 5.1(left) [18]. Its components are as shown in Figure 5.1(right). EDAR operates in purely event-driven manner in doing its parts' rearrangement tasks. It has two dimensional motion capacilities with encoder feedback as well as visual feedback of the workspace. Its workspace is restricted to a 2 m $\times$ 2 m area.

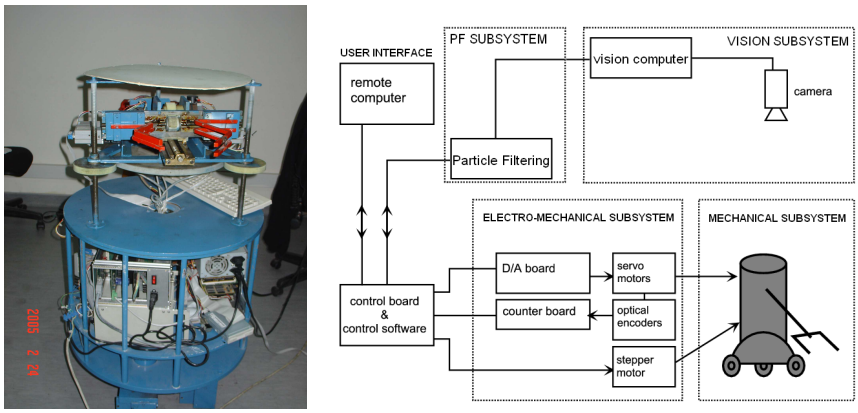

Figure 5.1. Left: EDAR robot; Right: System components

5.1. EDAR's Motion Capabilities

EDAR is a mobile robot with 2D linear and rotational motion capabilities, a three degrees of freedom arm and one degree of freedom gripper. These capabilities are realized through its electromechanical system consisting of six motors – five of which are dc servo motors and the last one is a stepper motor-, optical encoders, drivers, AD and DA boards. All the motion are controlled by servomotors except the stepper motor that enables 1 DOF grasping and ungrasping motion. Each servo motor has a two channel optical encoder which is also fed back to the robot. Detailed information is available from [18].

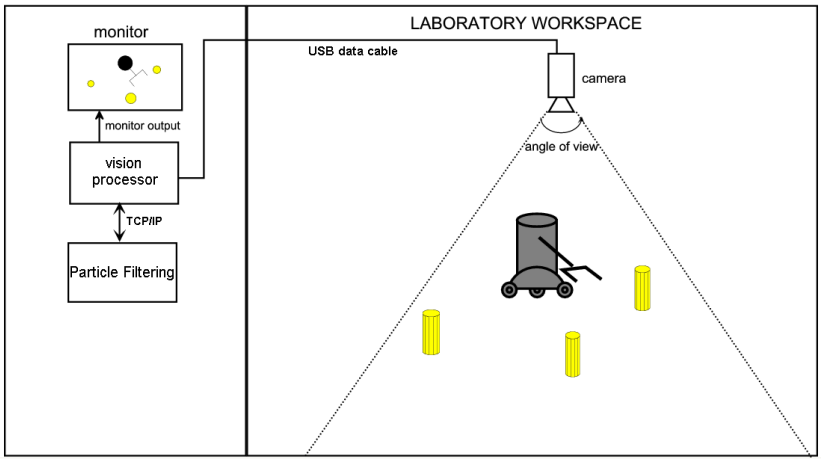

Figure 5.2. Vision system

5.2. Sensory Feedback – Visual Processing

EDAR incorporates a vision system which provides positional feedback regarding itself and those of all the parts. As depicted in Figure 5.2, a camera that is located above has a bird-eye view of its workspace and provides visual feedback. Through an orthographic projection, both the robot and the parts are mapped to a 2D workspace and are seen as disks with varying centers and radii. The vision system has been updated in order to provide this information in real-time. In its earlier version, an analog camera along with a DSP based vision processor board was used. This system was impeding EDAR's reactivity since its processing time was about 3 seconds. In order to overcome this problem, first, the analog camera has been replaced by USB camera that can then be connected directly to a mainboard. Next, the processing software has been completely rewritten in order to run on the mainboard. This change now provides an visual update of order 0.2 second.

The visual feedback is obtained based on the following visual processing stages[1] :

1. *Acquire_image:* A new color 24 bit 1280×1024 image is obtained from the USB camera. The image is then subsampled to an image of size 320×256. An sample

[1]As the focus is to provide reliable visual feedback to the robot with minimal effort, the visual processing is taken as simple as possible. Admittedly, a more elaborate scheme including the use of color may provide more powerful cues regarding the identity of the parts, etc.

image from a 3 part rearrangement task is as shown in Figure 5.3(a).

2. *Grey_level:* The color image is then converted to a gray image. Again, a sample resulting image is as shown Figure 5.3(b).
3. *Threshold:* A thresholding process is applied. Again, a sample resulting image is as shown in Figure 5.3(c).
4. *Edge_detection:* A high-frequency, spatial filter is applied in order to compute the edges. Again, a sample resulting image is as shown in Figure 5.3(d).
5. *Segmentation:* A connected component analysis is performed and a labeling algorithm segments the robot as well as all the parts in the workspace.
6. *Circle_fit:* For the robot and each part i, the corresponding circle is formed from each connected group of pixels using energy minimization. The energy function defined as $\sum_{j=1}^{N}(\|p_j - b_i\| - \rho)^2$ is minimized with respect to b_i. Here p_j denotes the position vector of the pixel j, N denotes the number of the pixels in the group, b_i and ρ_i denote the vector of the center point and the radius of the fitted circle respectively. This sum is minimized for each part in order to calculate the position and the radius information [18]. A similar approach is used to compute the robot's position and radius. Again a sample resulting image is as shown in Figure 5.3(e).
7. *Interpolation:* An interpolation method is used to convert the position and the radius data of the robot and the parts to the real workspace data.
8. *Send_data:* The positions and the radii of the robot and the parts in the workspace are then input to particle filtering. Detailed information is provided in the following section.

5.3. Particle Filter System

The PF code used in the simulation software is used with some modifications in this module. The PF system communicates with the vision system and EDAR via TCP and RS232, respectively. The flow of PF system is as shown in Figure 5.4.

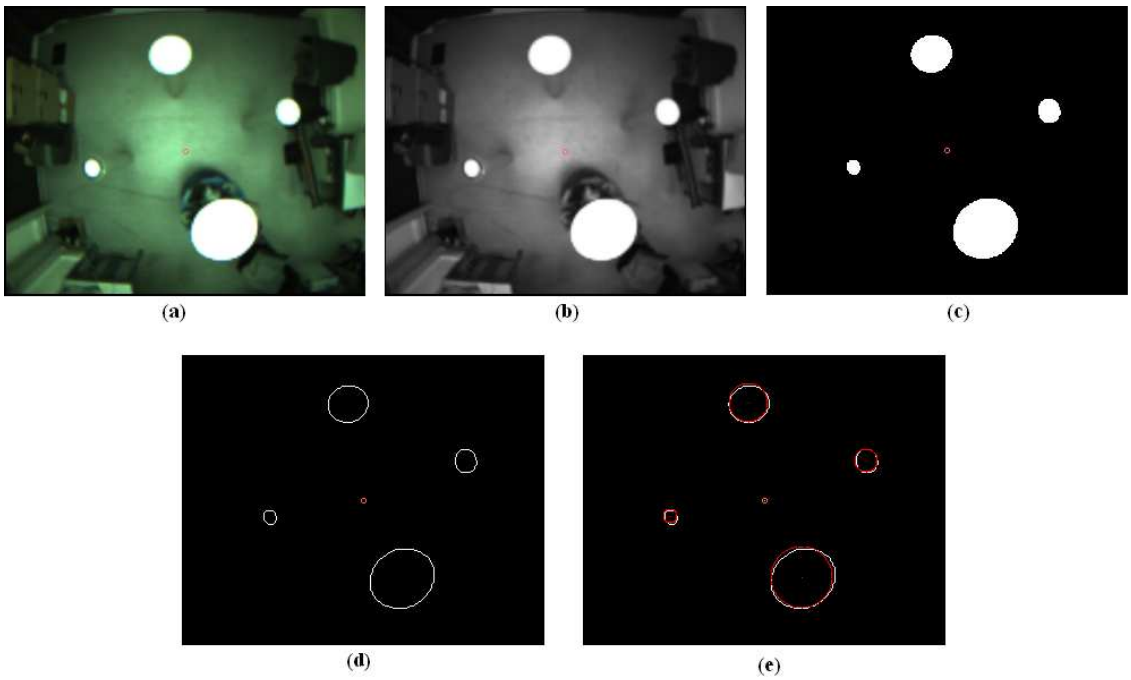

Figure 5.3. Visual processing stages: a) Subsampled color image, b) Gray image, c) Binary image, d) Edges, e) Fitted circles

5.4. Experimental Results

EDAR is engaged in 3-part rearrangement tasks with goal difficulties identical to those used in the 3-part simulations. The parts' radii vary from 6 cm to 11 cm. For each level of task complexity, the robot is made to run 5 sample tasks with random initial configurations. 100 particles are used in the PF system because of real-time constraints such as computational complexity. High noise is assumed in the measurements to cover the actual distribution.

Figure 5.5 presents the results of mean values of performance measures. It is observed that with the increasing task complexity the general trend of mean values of *nrl* and *npl* increases more than the other performance measures. The *nrl*, *npl* and *pi* values in the experiments using PF system are less than ones in the experiments done in the previous work [2]. The system noise level is between low and medium noise because the improvements in *rpee* and *ppee* are close to the ones in medium noise level. As to number of collisions, two collisions were occcured in hard task comlexity.

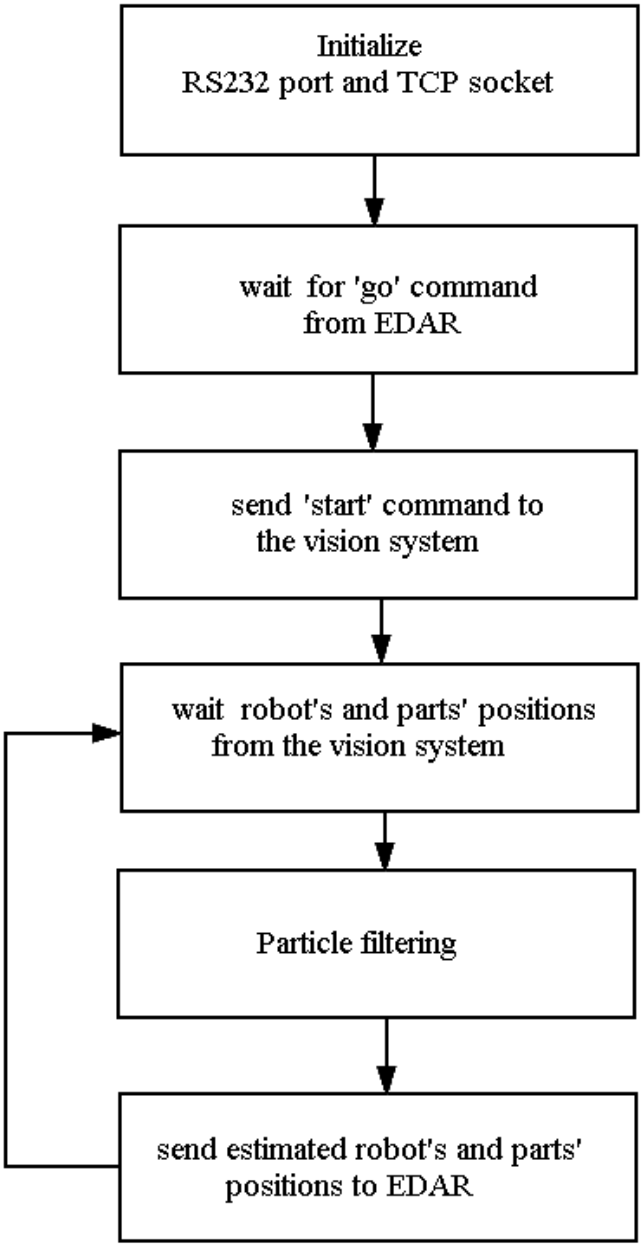

Figure 5.4. The flow of processing with particle filters

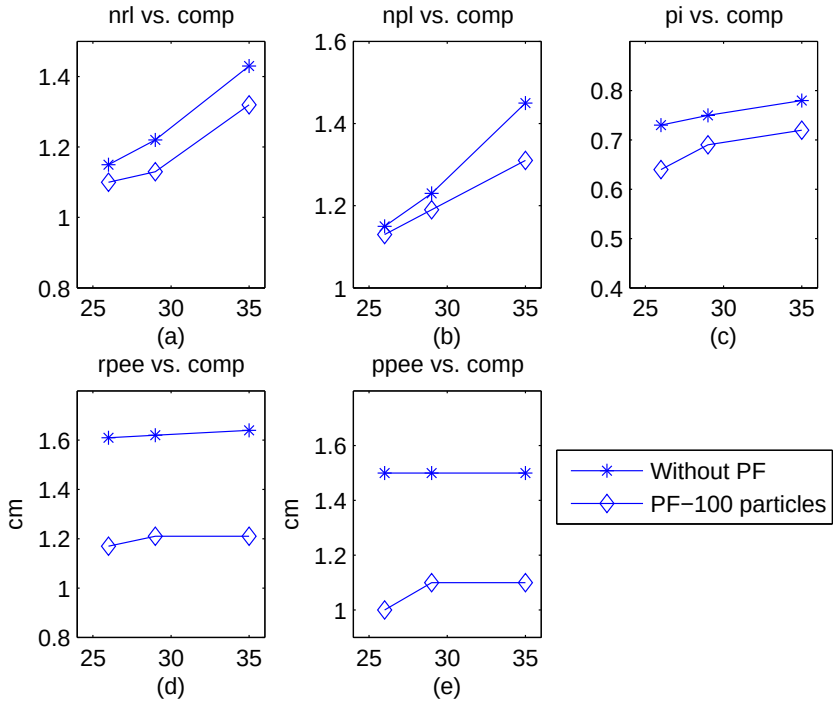

Figure 5.5. Performance measures vs. *comp* in the 3-part experiments

The experimental improvements in *nrl*, *npl*, *pi* and *rpee* are less than ones in the simulations. However, the improvement in the *ppee* is close to in the simulation. This is because parts' positions are not related to robot's dynamics.

EDAR accomplishes the moving task of three parts with a positional inaccuracy ranging over 10-15 cm/part without using PF. When using the PF, the *pi* value is ranging over 4-10 cm/part. This improvement stems from using PF and partly continuous visual feedback.

Using PF in EDAR provides improvements. The amount of improvements in the performance measures can be increased by improving the following factors: i) The inaccuracies in its linear and rotational movements as well the rotations of its arm; and ii) Inability to be actuated by torques lower than some predefined levels.

6. CONCLUSIONS

This thesis studies the feedback-based version of the parts' rearrangement problem under sensor inaccuracy. In this scenario, a robot inhabits the same workspace as a set of parts which need to be moved from an arbitrary initial placement to a final goal configuration. Unlike previous work, we no longer assume perfect sensory knowledge as will be the case in many applications. The models used for the dynamics and measurement are modified to include noise. As the resulting system has both nonlinear and linear parts, the states are estimated using particle and Kalman filters. The capability to handle nonlinear, non-Gaussian systems allows PFs to achieve improved accuracy over Kalman filter-based estimation methods.

The simulation results indicate that in cases of increased sensor inaccuracy, the robot's movements and the positional inaccuracy are both improved. More importantly, the number of collisions and the percentage of successful task completions are improved dramatically. In the analysis of the PF under different noise levels using CRLB, it is observed that with increased noise levels, the effect of the PF on reducing the error becomes more pronounced. The bound level in *move-part* stage is more rough than in *mate-part* stage. This is because the system dynamics in *move-part* stage is more complex than in *mate-part* stage.

In the 3-part experiments, performance measures are better than the ones in the experiments of previous work. Especially, the improvement in positional inaccuracy increases the accuracy of positioning a part in its goal configuration. The amount of improvements in the performance measures can be increased by improving EDAR's mechanical constraints.

For future work, since the computational power of EDAR is low, PF runs on the different computer which may cause communication delays. The computational power of EDAR can be increased. Thus, the PF runs on EDAR. Furthermore, after adding extra sensors such as sonar and laser sensors to EDAR, data fusion is done using PF.

APPENDIX A: PROOF OF RECURSIVE STATE ESTIMATION

Using the Markov property of $p(r_t|r_{t-1})$, $p(z_t|r_t)$ and Bayes' theorem

$$p(r|z) = \frac{p(z|r)p(r)}{p(z)}$$

$$p(z) = \int p(r,z)dr$$

It can be shown that

$$\begin{aligned} p(r_t|z_{0:t}) &= p(r_t|z_t, z_{0:t-1}) \\ &= \frac{p(z_t|r_t, z_{0:t-1})p(r_t|z_{0:t-1})}{p(z_t|z_{0:t-1})} = \frac{p(z_t|r_t)p(r_t|z_{0:t-1})}{p(z_t|z_{0:t-1})} \end{aligned}$$

Also,

$$\begin{aligned} p(z_t, r_t|z_{0:t-1}) &= p(z_t|r_t, z_{0:t-1})p(r_t|z_{0:t-1}) = p(z_t|r_t)p(r_t|z_{0:t-1}) \\ p(r_{t+1}, r_t|z_{0:t}) &= p(r_{t+1}|r_t, z_{0:t})p(r_t|z_{0:t}) = p(r_{t+1}|r_t)p(r_t|z_{0:t}) \end{aligned}$$

implying that (integrating w.r.t. r_t on both sides)

$$\begin{aligned} p(z_t|z_{0:t-1}) &= \int p(z_t|r_t)p(r_t|z_{0:t-1})dr_t \\ p(r_{t+1}, r_t|z_{0:t}) &= \int p(r_{t+1}|r_t)p(r_t|z_{0:t})dr_t \end{aligned}$$

Note that the conditional probability density functions $p(r_{t+1}|r_t)$ and $p(z_t|r_t)$ are known from system model.

APPENDIX B: SYSTEMATIC RESAMPLING

Using resampling can reduce the effects of degeneracy whenever a significant degeberacy is observed , that is, whenever N_{eff} falls below a certain threshold N_{th}. Resampling eliminates particles with low importance weights and multiplies paritcles with high importance weights (in apparent analogy to genetic algorithms). It involves generating a new set $\{r_t^{i*}\}_i^N$ with replacement N times from an approximate discrete representatin of $p(r_t|z_t)$.

Systematic resampling [9, 29] is simple to implement, its computational complexity $O(N)$ and it minimizes the MC variation. The particle selection process is schematically shown in Figure B.1, where acronym CSW stand for the cumulative sum of weights of random measure $\{r_t^i, w_t^i\}$, and random variable u_i, $i = 1, ..., N$ is uniformly distributed on the interval [0,1]. $u_i \sim U[0, 1]$ maps into index j; the corresponding particle r_t^j has a good chance of being selected and multiplied because of its high value of w_t^j.

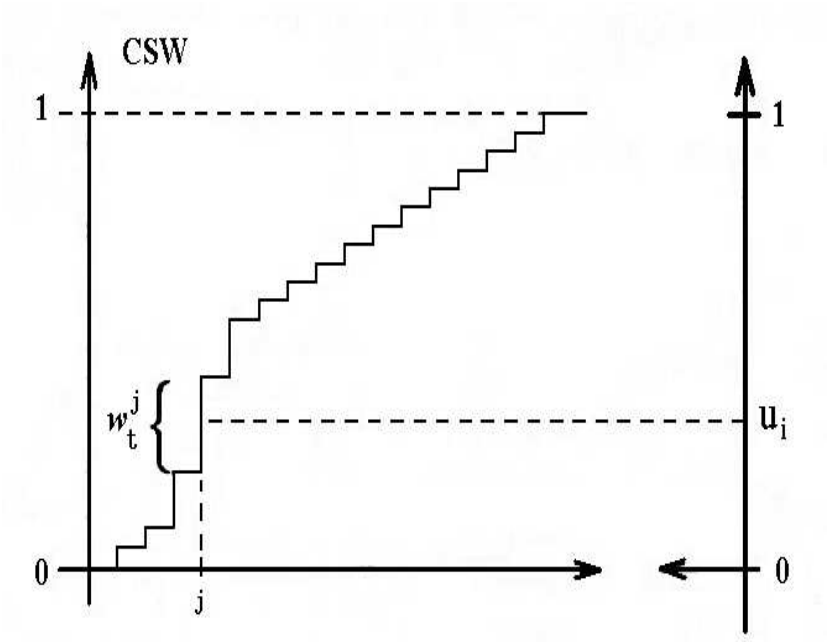

Figure B.1. The process of resampling

Table B.1. Systematic resampling algorithm

$\left\{r_t^{j^*}, w_t^j\right\}_{j=1}^{N} = RESAMPLE\left[\{r_t^i, w_t^i\}_{i=1}^{N}\right]$

- Initialize the CSW: $c_1 = w_t^1$
- FOR $i = 2 : N$
  - Construct CSW: $c_i = c_{i-1} + w_t^i$
- END FOR
- Start at the bottom of the CSW: $i = 1$
- Draw a starting point: $u_1 \sim U\left[0, N^{-1}\right]$
- FOR $j = 1 : N$
  - Move along the CSW: $u_j = u_1 + N^{-1}(j-1)$
  - WHILE $u_j > c_i$
    - $i = i + 1$
  - END WHILE
  - Assign particle: $r_t^{j^*} = r_t^i$
  - Assign weight: $w_t^j = w_t^i$
- END FOR
- Calculate total weight: $tot = SUM\left[\left\{w_t^j\right\}_{j=1}^{N}\right]$
- FOR $j = 1 : N$
  - Normalize: $w_t^j = w_t^j / tot$
- END FOR

The pseudocode of the systematic resampling algorithm is described in Table B.1 [10].

REFERENCES

1. Hopcroft, J. E., J. T. Schwartz and M. Sharir, "On the Complexity of Motion Planning for Multiple Independent Objects: PSPACE-hardness of the Warehouseman's Problem", *International Journal of Robotics Research*, Vol. 3, No. 4, pp. 76-88, 1984.

2. Karagöz, C. S., H. I. Bozma and D. E. Koditschek, "Feedback-Based Event-Driven Parts Moving", *IEEE Transactions on Robotics*, Vol. 20, No. 6, pp. 1012-1018, 2004.

3. Khatib, O., "Real Time Obstacle Avoidance for Manipulators and Mobile Robots", *International Journal of Robotics Research*, Vol. 5, No. 1, pp. 90-99, 1986.

4. Koditschek, D. E., "The Application of Total Energy as a Lyapunov Function for Mechanical Control Systems, in Control Theory and Multibody Systems", *American Mathematical Society*, pp. 131-158, 1989.

5. Rimon, E. and D. E. Koditschek, "Exact Robot Navigation Using Artificial Potential Functions", *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 5, pp. 501-518, 1992.

6. Bozma, H. I. and D. E. Koditschek, "Assembly as a Noncooperative Game of its Pieces: Analysis of 1D Sphere Analysis", *Robotica*, Vol. 19, pp. 93-108, 2001.

7. Cox, I. J. and G. T. Wilfong, *Autonomous Robot Vehicles*, Springer Verlag, 1990.

8. Doucet, A., S. Godsill and C. Andrieu, "On Sequential Monte Carlo Sampling Methods for Bayesian Filtering", *Statistics and Computing*, Vol. 10, No. 3, pp. 197-208, 2000.

9. Arulampalam, M. S., S. Maskell, N. Gordon and T. Clapp, "A Tutorial on Particle Filters for Online Non-linear/Non-Gaussian Bayesian Tracking", *IEEE Transactions on Signal Processing*, Vol. 50, No. 2, pp. 174-188, February 2002.

10. Rystic, B., S. Arulampalam and N. Gordon, *Beyond Kalman Filter Particle Filter for Tracking Applications*, Artech House, Boston, 2004.

11. Crisan, D. and A. Doucet, "A Survey of Convergence Results on Particle Filtering for Practitioners", *IEEE Transactions on Signal Processing*, Vol. 50, No. 3, pp. 736-746, 2002.

12. Dellaert, F., D. Fox, W. Burgard and S. Thurn, "Monte Carlo Localization for Mobile Robots", *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

13. Gustafsson, F., F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson and P. Nordlund, "Particle Filters for Positioning, Navigation and Tracking", *IEEE Transactions on Signal Processing*, Vol. 50, No. 2, pp. 425-437, 2002.

14. Montemerlo, M., S. Thurn and W. Whittaker, "Conditional Particle Filters for Simultaneous Mobile Robot Localization and People-Tracking", *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 695-701, 2002.

15. Kwok, C., D. Fox and M. Meila, "Real-time Particle Filters", *Neural Information Processing Systems*, 2002.

16. Karlsson, R., F. Gustafsson and T. Karlsson, "Particle Filtering and Cramer-rao Lower Bound for Underwater Navigation", *Proc. Acoustics, Speech, and Signal Processing(ICASSP '03)*, 2003.

17. Doucet, A., N. Freitas and N. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer, 2001.

18. Karagöz, C. S., *A Game Theoretic Approach to Objects' Moving Problem using Mobile Robots*, Ph.D. Thesis, Electrical and Electronics Eng., Bogazici University, 2001.

19. Gustafsson, F., F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, P.J. Nordlund, R. Karlsson, "A Framework for Particle Filtering in Positioning, Navigation and Tracking Problems", *Statistical Signal Processing, Proceedings of the 11th IEEE Signal Processing Workshop*, pp. 34-37, 2001.

20. Gordon, N. J., D. J. Salmond and A. F. M. Smith, "A Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation", *Proc. Inst. Elect. Eng. Radar, Sonar, Navigation*, Vol. 140, pp. 107-113, 1993.

21. Isard, M. and A. Blake, "Condensation-Conditional Density Propation- for Visual Tracking", *International Journal of Computer Vision*, pp. 5-28, 1998.

22. Carpenteri, J., P. Clifford and P. Fearnhead, "Improved Particle Filter for Nonlinear Problems", *IEE Proc. Part F, Radar and Sonar Navigation*, Vol. 146, pp. 2-7, 1999.

23. Liu, J. S. and R. Chen, "Sequential Monte Carlo Methods for Dynamical Systems", *J. Amer. Statist. Assoc.*, Vol. 93, pp. 1032-1044, 1998.

24. Haykin, S., *Adaptive Filter Theory*, Prentice-Hall, 4th ed., 2001.

25. Simandl, M., J. Kralovec and P. Tichavsky, "Filtering, Predictive, and Smoothing Cramer-rao Bouns for Discrete-time Nonlinear Dynamic Systems", *Automatica*, Vol. 37, pp. 1703-1716, 2001.

26. Bergman, N., *Recursive Bayesian Estimation: Navigation and Tracking Applications*, Ph.D. Thesis, Electrical Eng., Linköping Studies in Science and Technology, 1999.

27. Van Trees, H. L., *Detection, Estimation, and Modulation Theory*, John Wiley and Sons, 1968.

28. Tichavsky, P., C. H. Muravchik and M. Zakai, "Posterior Cramer-Rao Bounds for Discrete-time Nonlinear Filtering", *IEEE Transactions on Signal Processing*,

Vol. 46, pp. 1386-1396, May 1998.

29. Kitagawa, G., "Monte Carlo Filter and Smoother for Non-Gaussian Non-linear State space Models", *Journal of Computational and Graphical Statistics*, Vol. 5, No. 1, pp. 1-25, 1996.