Sequential Monte Carlo for Model Predictive Control

N. Kantas[†], J.M. Maciejowski[†] and A. Lecchini-Visintini[‡]

[†] *Cambridge University Engineering Dept., Cambridge, CB2 1PZ, UK*
{nk234,jmm}@cam.ac.uk
[‡] *Dept. of Engineering, University of Leicester, Leicester, LE1 7RH, UK,*
alv1@leicester.ac.uk

Keywords : Stochastic optimisation, Stochastic MPC, Sequential Monte Carlo

Abstract : **This paper proposes the use of Sequential Monte Carlo (SMC) as the computational engine for general (non-convex) stochastic Model Predictive Control (MPC) problems. It shows how SMC methods can be used to find global optimisers of non-convex problems, in particular for solving open-loop stochastic control problems that arise at the core of the usual receding-horizon implementation of MPC. This allows the MPC methodology to be extended to nonlinear non-Gaussian problems. We illustrate the effectiveness of the approach by means of numerical examples related to coordination of moving agents.**

1 Introduction

Nonlinear Model Predictive Control (MPC) usually involves non-convex optimisation problems, which in general suffer from the existence of several or even many local minima or maxima. This motivates the use of global optimisation algorithms, which guarantee asymptotic convergence to a global optimum. In most cases such algorithms employ a randomised search strategy to ensure that the search process is not trapped in some local mode. A popular example is Simulated Annealing (SA). Apart from the issue of multi-modalities of costs or rewards, solving such problems becomes even more complicated when stochastic processes are used to represent model uncertainties. In general, stochastic decision problems involve nonlinear dynamics with arbitrary distributions on general state spaces. In this paper we are mostly interested in continuous state spaces. Furthermore, the costs or rewards are usually expressed as expectations over relatively high-dimensional spaces. Monte Carlo methods are currently the most successful methods for evaluating such expectations under very weak assumptions, and have been widely applied in many areas such as finance, robotics, communications etc. An interesting point, which is overlooked often by the control community, is that Monte Carlo has also been applied for performing global optimisation, mainly in inference problems such as Maximum Likelihood or Maximum a Posteriori estimation, as presented recently in [1, 9, 13].

Still, solving stochastic optimal control problems on continuous state spaces for nonlinear non-Gaussian models is a formidable task. Solutions can be obtained by solving Dynamic Programming/ Bellman equations [3], but there is no analytical solution to this equation — except in very specific cases, such as finite state spaces or linear Gaussian state-space models with quadratic costs. In general, the value function takes as argument a probability distribution, and it is extremely difficult to come up with any sensible approximation to it. This is why, despite numerous potential applications, the literature on applications of Monte Carlo methods for control of non linear non Gaussian models is extremely limited [2].

MPC combined with Monte Carlo methods provides a natural approximation of solving the Bellman equation in the stochastic case, just as deterministic MPC can be viewed as a natural approximate method for solving deterministic optimal control

Int. Workshop on Assessment and Future Directions of NMPC
Pavia, Italy, September 5-9, 2008

problems [12]. For details of how MPC relates to dynamic programming and the Bellman equation, with emphasis on the stochastic case, see [4].

The most developed approaches for exploiting Monte Carlo methods for optimisation are based on either Markov Chain Monte Carlo (MCMC) methods [15], or Sequential Monte Carlo (SMC) methods [5, 7]. Considerable theoretical support exists for both MCMC and SMC under very weak assumptions, including general convergence results and central limit theorems [15, 5].

To date the control community has investigated the use of MCMC as a tool for evaluating approximate value functions, and SMC, in the guise of 'particle filters', for state estimation — see [14] for a setting closely related to MPC. Recently, in [10, 11] the authors proposed to use a MCMC algorithm similar to Simulated Annealing developed in [13], for sampling from a distribution of the maximisers of a finite-horizon open-loop problem, as the key component of an MPC-like receding-horizon strategy. As in any stochastic optimisation algorithm, the long execution times needed imply that these methods can be considered only for certain control problems, in which fast updates are not required. But even when restricted to such problems, the computational complexity of the algorithms can be very high. It is therefore important to take advantage of any structure that might be available in the problem. SMC seems to manage this better than MCMC in sequential problems. The computation can also be parallelised and requires less tuning than that required by standard MCMC algorithms.

In this paper we investigate the use of a Sequential Monte Carlo (SMC) approach, in contrast to the Markov chain Monte Carlo (MCMC) approach we proposed previously. This approach of using SMC methods for the sampling of global optimisers within MPC, is to the best of our knowledge novel. We propose some specific algorithmic choices in order to accelerate convergence of Simulated Annealing methods when applied to stochastic MPC problems. We shall demonstrate the effectiveness of our approach by means of numerical examples inspired by Air Traffic Management.

2 Problem Formulation

In general control problems one focuses on dynamical models, in which a specified user or controller or decision maker influences the evolution of the state, $X_k \in \mathcal{X}$, and the corresponding observation, $Y_k \in \mathcal{Y}$, by means of an action or control input, $A_k \in \mathcal{A}$, at each time k. Consider the following nonlinear non-Gaussian state space model

$$X_{k+1} = \psi\left(X_k, A_{k+1}, V_{k+1}\right), \; Y_k = \phi\left(X_k, A_k, W_k\right),$$

where $\{V_k\}_{k\geq 1}$ and $\{W_k\}_{k\geq 0}$ are mutually independent sequences of independent random variables and ψ, ϕ are nonlinear measurable functions that determine the evolution of the state and observation processes. The decision maker tries to choose the sequence $\{A_k\}_{k\geq 0}$, so that it optimises some user specified sequence of criteria $\{J_k\}_{k\geq 0}$.

In this paper we shall restrict our attention to the fully observed case ($Y_k \equiv X_k$), although our results can be generalised for the partially observed case as well. Furthermore, as our goal is to develop an algorithm for use with MPC, we will focus only on finite horizon problems. We refer the interested reader to [2] for a treatment on how SMC has been used for the infinite horizon case using stochastic gradients instead.

Conditional upon $\{A_k\}_{k\geq 0}$, the process $\{X_k\}_{k\geq 0}$ is a Markov process with $X_0 \sim \mu$ and Markov transition density $f\left(x' \middle| x, a\right)$, so that we can write

$$X_{k+1} \middle| \left(X_k = x, A_{k+1} = a\right) \sim f\left(\cdot \middle| x, a\right). \tag{1}$$

These models are also referred to as Markov Decision Processes (MDP) or controlled Markov Chains.

We will now formulate an open loop problem solved at each MPC iteration. Let us introduce a measurable reward function $h : \mathcal{X} \times \mathcal{A} \to \mathbb{R}_+$, for the following additive reward decision problem. At time $k-1$, the action A_{k-1} has been selected, the state

X_{k-1} is measured and then at time k one wants to maximise the function J_k defined as

$$J_k\left(A_{k:k+H-1}\right) = \mathbb{E}\left[\sum_{n=k}^{k+H-1} h\left(X_n, A_n\right)\right], \tag{2}$$

where $A_{k:k+H-1}$ denotes the joint vector $A_{k:k+H-1} = [A_k, ..., A_{k+H-1}]$ and the expectations are with respect to the joint distribution of the states $p_{A_{k:k+H-1}}(x_{k:k+H-1})$, giving

$$J_k\left(A_{k:k+H-1}\right) = \int_{\mathcal{X}^H} \underbrace{\left(\sum_{n=k}^{k+H-1} h\left(x_n, A_n\right)\right)}_{u(A_{k:k+H-1}, x_{k:k+H-1})} \underbrace{\left(\prod_{n=k+1}^{k+H-1} f\left(x_n \middle| x_{n-1}, A_n\right)\right) f\left(x_k \middle| X_{k-1}, A_k\right)}_{p_{A_{k:k+H-1}}(x_{k:k+H-1})} dx_{k:k+H-1}, \tag{3}$$

where we define

$$u(A_{k:k+H-1}, x_{k:k+H-1}) = \sum_{n=k}^{k+H-1} h\left(x_n, A_n\right). \tag{4}$$

We aim to perform the following maximisation

$$A^*_{k:k+H-1} = \arg \max_{A_{k:k+H-1} \in \mathcal{A}^H} J_k\left(A_{k:k+H-1}\right),$$

in order to obtain a solution for the open loop problem.

Of course, this is not a trivial task. If the control input took its values in a finite set $\mathcal{A}$ of cardinality K, it would be possible to approximate numerically this cost using particle methods or MCMC for the K^H possible values of $A_{k:k+H-1}$ and then select the optimal value. In [2] the authors present in detail how to get a particle approximation of J_k using standard SMC results for filtering. Of course, in practice such an approach cannot handle large values of H or K. Moreover if A_k takes values in a continuous space $\mathcal{A}$ and $J_k\left(A_{k:k+H-1}\right)$ is differentiable with respect to $A_{k:k+H-1}$, one can still resort to a gradient search in $\mathcal{A}^H$. This has been presented in [2]. Using gradients would imply, as in any local optimisation method, that multiple runs from different initial points are needed to get a better estimate of the global optimum, but still it is difficult to get any formal guarantees. This motivates the use of Monte Carlo optimisation.

3 Monte Carlo Optimisation

Maximising (3) falls into the broader class of problems of maximising

$$J\left(\theta\right) = \int_{\mathcal{Z}} u(\theta, z) p_\theta(z) dz, \tag{5}$$

where we define $\theta = A_{k:k+H-1}$ and $z = x_{k:k+H-1}$, while θ^* are the maximisers of J. In this section we show how Monte Carlo simulation can be used to maximise J. In [1, 13] MCMC algorithms have been proposed for this and in [10] the authors explained how they can be combined with MPC. More recently, in [9] SMC methods have been applied for solving a marginal Maximum Likelihood problem, whose expression is similar to (5). In the remainder of this paper we shall focus on deriving a similar algorithm to [9], intended to be used for MPC. The main difference of our approach is that our problem formulation exhibits a slightly different structure. In fact, we are using a dynamical model intended for control problems, and therefore are doing inference to compute time varying optimal actions instead of static parameters, which is the purpose of parameter estimation. Although the difference seems subtle at first glance, it is important and leads to similar algorithms showing completely different behaviour.

The basic idea is the same as in [1, 9, 13]. First we assume $u(\theta, z)$ is nonnegative. Note that this might seem restrictive at the beginning but we remark that any

maximisation remains unaffected with respect to shifting by some finite positive constant. As in the standard Bayesian interpretation of Simulated Annealing, we define a distribution $\widetilde{\pi}_\gamma$

$$\widetilde{\pi}_\gamma(\theta) \propto p(\theta) J(\theta)^\gamma ,$$

where $p(\theta)$ is an arbitrary prior distribution, which contains the maximisers θ^* and encapsulates any prior information on θ not captured by the model. As such information is not likely to be available, uninformative priors might be used. Under weak assumptions, as $\gamma \to \infty$, $\widetilde{\pi}_\gamma(\theta)$ becomes concentrated on the set of maximisers of J [8].

We now introduce γ artificial replicates of z, all stacked into a joint variable $z_{1:\gamma}$ and define the distribution π_γ

$$\pi_\gamma(\theta, z_{1:\gamma}) \propto \prod_{i=1}^{\gamma} u(\theta, z_i) p_\theta(z_i).$$

It easy to show that the marginal of π_γ is indeed proportional $\widetilde{\pi}_\gamma$, i.e.

$$\widetilde{\pi}_\gamma(\theta) \propto \int \pi_\gamma(\theta, z_{1:\gamma}) dz_{1:\gamma}.$$

We now define a strictly increasing integer infinite sequence $\{\gamma_l\}_{l\geq 0}$, which will play the role of the inverse temperature (as in SA). For a logarithmic schedule one can obtain formal convergence results [13]. For a large l, the distribution $\pi_{\gamma_l}(\theta, z_{1:\gamma_l})$ converges to the uniform distribution of the maximisers of J, [8]. In practice logarithmic schedules lead to slow convergence; more quickly increasing rates and finite sequences $\{\gamma_l\}_{l\geq 0}$ are therefore used. In general it is impossible to sample directly from π_γ, hence various Monte Carlo schemes have been proposed. In [1, 13] this is achieved by MCMC, and in [9] an SMC sampling approach was proposed for a Maximum Likelihood problem, based on the generic SMC algorithm found in [6]. The SMC approach can achieve more efficient sampling from π_γ, and avoids some of the fundamental bottlenecks of MCMC-based optimisation.

4 Stochastic Control using MPC based on SMC

SMC is a popular technique, applied widely in sequential inference problems. The underlying idea is to approximate a sequence of distributions $\pi_l(x)$[1] of interest as a collection of N discrete masses of the variables (also referred as particles $\{X_l^{(i)}\}_{i=1}^N$), properly weighted by a collection of weights $\{w_l^{(i)}\}_{i=1}^N$ to reflect the shape of the distribution π_l. As π_l can be time varying, the weights and the particles are propagated iteratively by using a sequential importance sampling and resampling mechanism, which uses the particles of iteration $l-1$ to obtain new particles at iteration l. We shall be referring to $\{X_l^{(i)}, w_l^{(i)}\}_{i=1}^N$ as the particle approximation $\widehat{\pi}_l$ of π_l and this should satisfy

$$\sum_{i=1}^{N} w_l^{(i)} \delta_{X_l^{(i)}}(dx) \overset{N\to\infty}{\underset{a.s.}{\to}} \pi_l(dx),$$

where δ is a Dirac delta mass. For more details, see [5, 6, 7]. In Figure 1, we set out an SMC algorithm which can be used for the MPC problem defined in Section 2.

Steps I) 1-3 of the algorithm are iterated recursively to obtain a particle approximation for the maximisers of J_k. Referring to the general description of SMC in the previous paragraph, one can associate π_{γ_l} with π_l. We shall be using iteration number l, to index the propagation of π_{γ_l}. As we cannot run an infinite number of iterations, we shall terminate the iteration at $l = l_{\max}$. Note that l should not be confused with the time index k of Section 2 regarding the real time evolution of the state. To avoid this, we define $\theta_k = A_{k:k+H-1}$ and $z_k = x_{k:k+H-1}$, and also add

[1] x is not meant to be confused with x_k. Later it will be apparent that we shall be using $(\theta_k, z_{k,1:\gamma_l})$ as the variable of interest.

`At time` k,

- I) `For` $l = 1, ..., l_{\max}$:

1. `Sampling new particles:`
   - `For each particle` $i = 1, ..., N$ `sample`: $A^{(i)}_{k:k+H-1,l} \sim q_l(\cdot|X^{(i)}_{k:k+H-1,1:\gamma_{l-1}}, A^{(i)}_{k:k+H-1,l-1})$
   - `For each particle` $i = 1, ..., N$ `sample replicas of the joint state trajectory, for` $j = \gamma_{l-1} + 1, ..., \gamma_l$, $X^{(i)}_{k:k+H-1,j} \sim \prod_{n=k}^{k+H} f(x_n|x_{n-1}, A^{(i)}_{n,l})$.
2. `Weighting particles: for each particle` $i = 1, ..., N$ `assign weights` $w^{(i)}_l = w^{(i)}_{l-1} \prod_{j=\gamma_{l-1}+1}^{\gamma_l} u(A^{(i)}_{k:k+H-1,l}, X^{(i)}_{k:k+H-1,j})$, `normalise` $w^{(i)}_l = \frac{w^{(i)}_l}{\sum_{j=1}^{N} w^{(j)}_l}$.
3. `Resample, if necessary, to get new particle set` $\{(A^{(i)}_{k:k+H-1,l}, X^{(i)}_{k:k+H-1,1:\gamma_l})\}_{i=1}^N$ `with equal weights` $w^{(i)}_l = \frac{1}{N}$.

- II) `Compute the maximiser estimate` $\widehat{A}_{k:k+H-1}$
- III) `Apply` $\widehat{A}_k$ `as the action of time` k.

`Obtain measurement` $Y_k = X_k$ `and proceed to time` $k+1$

Figure 1: SMC Algorithm for MPC

a subscript k to π_γ to show the real time index. At each epoch k , we are interested in obtaining $l_{\max}$ consecutive particle approximations of $\pi_{k,\gamma_l}(\theta_k, z_{k,1:\gamma_l})$, where $z_{k,1:\gamma_l} = [z_{k,\gamma_1}, ..., z_{k,\gamma_l}]$[2]. At each iteration l, we obtain particle approximations $\widehat{\pi}_{\gamma_l}$, $\{(\Theta^{(i)}_{k,l}, Z^{(i)}_{k,1:\gamma_l}),\ w^{(i)}_l\}_{i=1}^N$, by propagating the particles of the previous approximation $\widehat{\pi}_{k,\gamma_{l-1}}$, $\{(\Theta^{(i)}_{k,l-1}, Z^{(i)}_{k,1:\gamma_{l-1}}), w^{(i)}_{l-1}\}_{i=1}^N$, weighting the new particles and then resampling.

We now explain briefly how steps 1 to 3 can be derived. Suppose we are at epoch k and iteration l. For the sampling step, we assume in this paper that we can sample from the model of the state, $p_{\theta_k}(z_k)$, by repeatedly sampling from each transition density f. This is not always possible, but for most practical control problems it is. If one cannot sample directly from f then importance sampling can be used. For every particle i, to get a sample $Z^{(i)}_{k,j} = X^{(i)}_{k:k+H-1,j}$, we use the previous measured state X_{k-1} and then repeatedly sample $X^{(i)}_{n,j} \sim f(\cdot|X^{(i)}_{n-1,j}, A^{(i)}_{n,l})$ for $n = k, ..., k+H-1$. For sampling new particles $\Theta^{(i)}_{k,l}$, an importance sampling approach has to be used at each l. We shall be using an importance distribution q_l to obtain $\Theta^{(i)}_{k,l} \sim q_l(\cdot|Z^{(i)}_{k,1:\gamma_{l-1}}, \Theta^{(i)}_{k,l-1})$ by simulation. We have intentionally chosen q_l to be varying with l and to depend on $Z_{k,1:\gamma_l}$ as this is more convenient for the general design setting. We shall not provide details on how to design q_l, as this depends on the problem specifics [7]. We shall refer the reader again to [5, 6] for a more general treatment.

For the weighting step we use

$$\frac{\pi_{k,\gamma_l}}{\pi_{k,\gamma_{l-1}}} \propto \prod_{i=\gamma_{l-1}+1}^{\gamma_l} u(\theta_k, z_{i,k}) p_{\theta_k}(z_{k,i})$$

[2]Note that $z_{k,1:\gamma_l}$ is the stacked vector of the γ_l artificial replicates of $x_{k:k+H-1}$ and n is used as a sequence index for the interval $k : k + H - 1$.

to obtain $\frac{w_l^{(i)}}{w_{l-1}^{(i)}}$ as an importance ratio proportional to $\prod_{j=\gamma_{l-1}+1}^{\gamma_l} u(\Theta_{k,l}^{(i)}, Z_{k,j}^{(i)})$. To obtain $u(\Theta_{k,l}^{(i)}, Z_{k,j}^{(i)})$ – see (3) and (4) – one can evaluate $h\left(X_{n,j}^{(i)}, A_{n,l}^{(i)}\right)$ point-wise at each step n during the sampling stage, and then get the total value of $u(\Theta_{k,l}^{(i)}, Z_{k,j}^{(i)})$. After normalising the weights one can perform a resampling step according to the multinomial distribution $\{(\Theta_{k,l}^{(i)}, Z_{k,1:\gamma_l}^{(i)}), w_l^{(i)}\}_{i=1}^N$, if the variance of the weights is low; see [7] for details.

As as regards step II, after having obtained the particle approximation $\widehat{\pi}_{k,\gamma_{l_{\max}}}$,one could use in principle any sample $\Theta_{k,l_{\max}}^{(i)}$ from the final particle set as the estimator of θ_k^*, where θ_k^* is the maximiser of $J_k(\theta_k)$. Given that $\widehat{\pi}_{k,\gamma_{l_{\max}}}$ should converge close to a uniform distribution over the set of θ_k^*, then any sample $\Theta_{k,l_{\max}}^{(i)}$ should be sufficiently close to θ_k^*. To improve things one could either choose a sample randomly according to its weight as in resampling, or in a deterministic fashion, by choosing the sample with the maximum weight. In many cases, such as if there is some symmetry in the location of the maximisers, this should be much better than using the empirical mean $\sum_{i=1}^N w_{l_{\max}}^{(i)} \Theta_{k,l_{\max}}^{(i)}$ to compute $\widehat{\theta}_k$.

We can use this open loop solution for performing an MPC step at step III. Once $\widehat{A}_{k:k+H-1}$ is computed, we then apply $\widehat{A}_k$. Then we proceed to time $k+1$ and repeat steps I-III for optimising J_{k+1}.

5 Numerical Examples

In this section we demonstrate how the proposed algorithm can be used in navigation examples, where it is required to coordinate objects flying at constant altitude, such as aircraft, UAVs, etc. We consider a two-dimensional constant speed model for the position of an object controlled by changing its bearing

$$X_{k+1} = X_k + v\tau[\sin\phi_{k+1}, \cos\phi_{k+1}]^T + b_{k+1} + V_{k+1}, \tag{6}$$

where v is the speed of the object, τ is a measuring period, ϕ is the bearing , b_n represents the predicted effect of the wind and $V_k \overset{iid}{\sim} \mathcal{N}(0, \Sigma)$. Although this is a linear kinematic model with Gaussian added noise, the algorithm in Figure 1 can handle nonlinear and non-Gaussian cases as it requires no assumptions on the dynamics or distributions.[3] We shall be using some way points α_n that the object is desired to pass through at each time n. We shall encode this in the following reward at time k,

$$J_k(\phi_{k:k+H-1}) = \mathbb{E}[\sum_{n=k}^{k+H} (c - \|X_n - \alpha_n\|_Q^2 - \|\phi_n - \phi_{n-1}\|_R^2)],$$

where $c > 0$ is sufficiently large to ensure $c - \|X_n - \alpha_n\|_Q^2 - \|\phi_n - \phi_{n-1}\|_R^2 \geq 0$, and $Q, R \geq 0$ are matrices of appropriate sizes.

We shall be investigating a number of scenarios. Firstly assume there are three way-points to be cleared, such that $\alpha_1 = \alpha_2 = ... = \alpha_{H_1}$, $\alpha_{H_1+1} = ... = \alpha_{H_2}$ and $\alpha_{H_2+1} = ... = \alpha_H$. If a single object obeying (6) starts at some initial position, then choosing a maneuver to maximise J_k means that it should pass through the points and stay near the check points as long as possible. The result of applying the algorithm of Figure 1 is shown in Figure 2(a).

We proceed by adding additional objects that also obey the dynamics of (6). Suppose that safety requirements impose the constraint that objects should not come closer to each other than a distance d_{min}. This makes the problem much harder as one has to ensure that constraints are not violated and the constraints have a significant effect on the multi modality of the reward. Let X_k^j denote the position of the jth

[3] Also non-convex constraints can be added.

object. The feasible space $\mathcal{X}^H$ is modified so that $\mathcal{X}^H = \{X_n^j \in \mathbb{R}^2 : \left\|X_n^j - X_n^i\right\| \geq d_{min}, \forall i \neq j, n = k, ..., k + H - 1\}$. Moreover, all expectations presented so far, for example equation (3), should be defined over the new feasible spaces for each object. To account for this we could modify the instantaneous reward to $h\left(x_n^j, A_n^j\right) \mathbf{1}_{X_{0:n}^j \in \mathcal{X}^H}$, where $\mathbf{1}_{x \in B}$ is an indicator function for the set B. Such a simple penalty approach means that no reward should be credited if a sampled trajectory $Z_{k,j}^{j,(i)}$ does not obey the constraint and its corresponding weight should be set to zero. This is a simulation based approach to deal with constraints, i.e. we propagate state samples only from the feasible region of the state space. We also assume that the initial condition of the system is not in violation of the constraint. Then, given that the SMC optimisation algorithm uses a large number of particles and samples many replicates of the state trajectories, this should allow safe decisions to be made. For a finite number of samples one could also obtain expressions for the probability of violating a constraints, e.g. using the Chebychev inequality. In practice, when using large number of particles the violation probability was observed to be very low (10^{-7}).

We have verified this using two different scenarios. In the first one seen in Figure 2(b), two objects flying in parallel towards the same direction, try to approach parallel way-points. MPC was used for repeated number of runs and no conflict between two objects took place. Further scenarios are depicted in Figures 2(c) and 2(d). These show a more complicated problem, in which four objects are directed towards each other and their way-points would lead them to a collision if constraints were not taken into account. In Figure 2(c) we plot the open loop solution of the problem at time $k = 1$ for a random disturbance sequence and in Figure 2(d) the MPC solution. We see that three objects try to cruise as closely as possible between their way-points and orbit each way point for some time, while one orbits waiting for the others. Again no conflict took place in multiple runs.

As a concluding remark, we would like to stress that little tuning was done to produce the results shown here. Also, we have not used any structural properties of Linear Gaussian systems with quadratic costs and just implemented the algorithm in Figure 1. The examples show early results from ongoing work, but they already demonstrate that the proposed SMC algorithm can be effective for non-convex decision problems.

Acknowledgements

This work was supported by EPSRC, Grant EP/C014006/1, and by the European Commission under project iFly FP6- TREN-037180. The authors are also grateful to Zhe (Anna) Yang for providing recommendations to improve the quality of the paper.

References

[1] Amzal B., Bois F.Y., Parent E., Robert C.P. (2006). Bayesian-Optimal Design via Interacting Particle systems. *Journal of the American Statistical Association*, Vol 101, No. 474, Theory and Methods.

[2] Andrieu C., Doucet A., Singh S.S., Tadić V., (2004) Particle Methods for Change Detection, Identification and Control, *Proceedings of the IEEE*, vol. 92, pp. 423 - 438

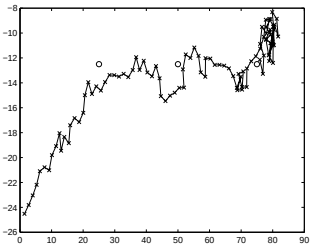
[3] Bertsekas D.P. (1995) *Dynamic programming and optimal control.* Belmont: Athena Scientific.

[4] Bertsekas D.P.(2005) Dynamic Programming and Suboptimal Control: A Survey from ADP to MPC, *European J. of Control,* Vol. 11, Nos. 4-5, 2005

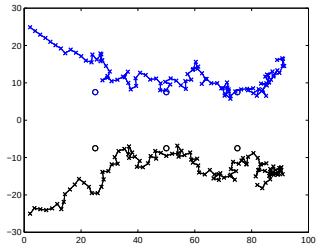
[5] Del Moral P. (2004). *Feynman-Kac formulae: genealogical and interacting particlesystems with applications.* New York: Springer Verlag.

[6] Del Moral P., Doucet A., Jasra A. (2006) Sequential Monte Carlo Samplers, *J. Royal Statist. Soc. B*, vol. 68, no. 3, pp. 411-436.

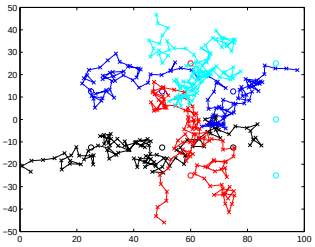
[7] A. Doucet, J.F.G. de Freitas and N.J. Gordon, *Sequential Monte Carlo Methods in Practice.* New York: Springer, 2001.



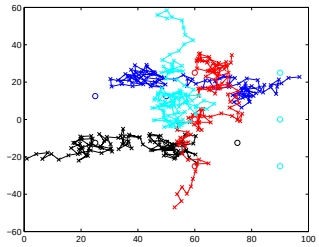
(a) Single-object example of MPC



(b) Two objects flying parallel using MPC



(c) Four-object example open loop solution at $k = 1$



(d) Four-object example using MPC

Figure 2: Application of SMC optimisation for different scenarios. A linear inverse temperature schedule was used, $\gamma_l = 5 + 3l$ and $l_{max} = 100$. For the rest of the parameters we used $N = 300$, $T = 100$ $v\tau = 2$, $b_k = [0.2, 0.2]^T$, $\Sigma = I$, $d_{min} = .5$, $Q = .1$, and $R = .01$. The waypoints of each object are plotted with circles with the same colour as the track of the object. When implemented in Intel Core Duo T2250 at 1.73GHz processor without any parallelisation, the computational time for each k is .05 sec per H per N per plane.

[8] Hwang, C.R.(1980) Laplaces method revisited: weak convergence of probability measures. *Ann. Probab.* 8(6), 11771182.

[9] Johansen A. M., Doucet A., Davy M. (2008). Particle methods for Maximum Likelihood Parameter Estimation in Latent Variable Models, *Statistics and Computing*, 18(1):47-57, 2008.

[10] Lecchini Visintini A., Glover W., Lygeros J., Maciejowski J.M. (2006) . Monte carlo optimization for conflict resolution in air traffic control. *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp 470-482.

[11] Lecchini Visintini A., Lygeros J., Maciejowski J.M. (2008) Simulated Annealing: Rigorous finite-time guarantees for optimization on continuous domains, In J.C. Platt, D. Koller, Y. Singer, S. Roweis, editors, *Advances in NIPS 20*, 865–872. MIT Press.

[12] Maciejowski J.M. (2002) *Predictive control with Constraints*, Prentice Hall.

[13] Muller P., Sanso B. G., De Iorio M. (2004). Optimal Bayesian design by Inhomogeneous Markov Chain Simulation. *Journal of the American Statistical Association*, pp. 788-798.

[14] Rawlings, J. B. and Bakshi B. R.(2006) Particle filtering and moving horizon estimation. *Comput. Chem. Eng.*, 30:15291541.

[15] Robert, C. P. and Casella, G. (2004) *Monte Carlo Statistical Methods.* 2nd Ed., Springer.