TOWARDS OPTIMAL CONTROL OF SELF-ORGANIZED ROBOTIC INSPECTION SYSTEMS

Nikolaus Correll and Alcherio Martinoli

Swarm-Intelligent Systems Group
École Polytechnique Fédérale de Lausanne, Switzerland
[nikolaus.correll|alcherio.martinoli]@epfl.ch

Abstract: We consider a swarm-intelligent inspection system concerned with the inspection of blades in a jet turbine. The system is based on a swarm of autonomous, miniature robots, using only on-board, local sensors. We capture the dynamics of the system at a higher abstraction level using non-spatial, probabilistic, discrete-time macroscopic models, which we use in an optimal control framework to find an optimal collaboration policy minimizing time to completion and overall energy consumption of the swarm. We consider time-invariant and time-variant decision variables for various stage constraints (energy consumption), and find optimal profiles using an extremum-seeking control scheme. In particular, we show that using a communication-based policy exclusively towards the end of the inspection progress decreases time to task completion, but only if there are at least as many robots as blades.

1. INTRODUCTION

For designing self-organized robotic systems, understanding of the relation between individual and collective behavior is of outmost importance. To avoid costly and time-consuming experiments, to allow for a priori insight into a given system, and to formally address stability and convergence properties, appropriate models are necessary.
Self-organization is emerging from the interplay of four ingredients. Positive feedback (e.g., amplification) and negative feedback (e.g., saturation, resource exhaustion), randomness, and multiple interactions among individuals (Bonabeau *et al.*, 1999). While self-organization might achieve less efficient coordination than other distributed control schemes, it can provide extremely high levels of robustness and can be applied to miniature robotic platforms such as those mentioned in this paper.
Distributed coordination policies for coverage and collective navigation tasks using multi-agent systems have been formally assessed using methods from automatic control by, for instance, Cortés *et al.* (2004), and Jadbabaie *et al.* (2003), respectively. The applied (deterministic) models however usually are — while being mathematically rigorous — not of easy application to real robotic swarms characterized by limited computation and communication capabilities as well as noisy sensors and actuators. Furthermore, Milutinovic *et al.* (2003) have applied optimal control theory in combination with macroscopic models based on hybrid automata in order to achieve central coordination of a robotic swarm. Being randomness and fully distributed control at the core of a self-organized swarm coordination, none of these modeling approaches was well suited for our purposes. For this reason, we make use of probabilistic macroscopic models, able to statistically capture the average dynamics and performances of a self-organized robotic swarm. The model has been developed based on a incremental, multi-level modeling methodology (realistic, microscopic and macroscopic models) which has

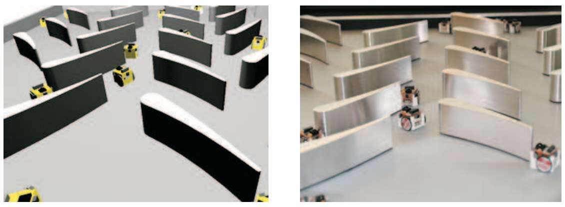

Fig. 1. *Left*: Overview of the turbine set-up in the realistic simulator. *Right*: Overview of the real-robot set-up.

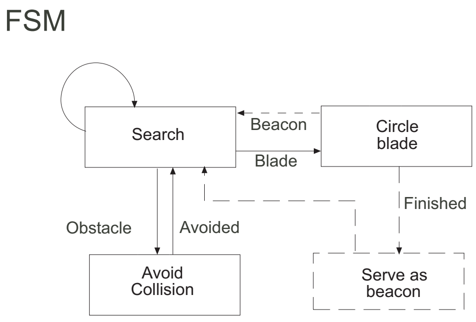

Fig. 2. The high-level behavioral flowchart of the robot controller as a Finite State Machine (FSM).

been successfully applied to several case studies (see for instance, Martinoli *et al.*, 2004), and which has lead to quantitative good agreement between reality and modeling predictions. In this contribution, we move a step forwards and we show how this type of models can be useful for dynamic optimization (Kirk, 2004) of the individual control parameters of a homogeneous swarm engaged in the inspection of a jet-turbine interior (Correll and Martinoli, 2006*a*).

As we showed earlier for this case study, predictions of the macroscopic model could be validated experimentally (realistic simulation and robotic experiments, see Figure 1) for some particular experiments, and modeling was hence used to explore different parameters of the individual robot controllers by systematically searching the design space (Correll and Martinoli, 2006*b*). In this work we will consider dynamic beacon-policies as we introduced them in (Correll and Martinoli, 2005) together with terminal constraints on the consumed total energy, which renders systematically searching the design space extremely time consuming.

1.1 Case study and robots' controllers

The overall behavior of a robot in a beacon-less policy (no communication) can be summarized as follows (see Fig. 2, without dashed states). The robot searches for blades throughout the turbine, combining schemes that drive the robot forward, avoid obstacles, and follow contours. Team-mates, walls, and blades are differentiated by on-board sensors, and are avoided or inspected, respectively.

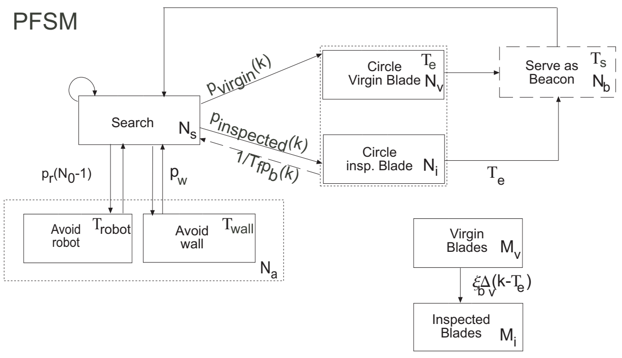

Fig. 3. The corresponding Probabilistic FSM (PFSM) used in the models, capturing details of interest of the schema-based controller. Dashed arrows/states characterize the beacon-based behavior.

A robot can start circumnavigating a blade at any point of its contour but can leave the blade exclusively at its tip. This allows the robot to bias its blade-to-blade trajectory without using any sophisticated navigation mechanisms, exploiting a specific feature of the environmental pattern (Correll and Martinoli, 2006*a*). A blade can be left only if a timeout parameter T_{max} has expired. The corresponding timer is set when the robot attaches to the blade. In (2005), Correll and Martinoli add the following behavior (see Fig. 2, dashed states and arrows): after inspecting a blade, a robot remains at its tip for the time T_s, where it serves as a beacon and signals to all robots within its range possibly approaching this blade to avoid it, or abort its circumnavigation, in case the other robot has already attached to the blade at another point. We note that in addition to the mere interaction based on mutual avoidance, signaling introduces a further coupling mechanism among the robots, which is in this case the main positive feedback leading to spatial self-organization.

2. MACROSCOPIC MODELS FOR SWARM ENGINEERING

The central idea behind our macroscopic models is to describe the experiment as a series of rate equations whose parameters are computed from the interactions' geometrical properties and systematic experiments with one or two real robots or realistic simulation (Martinoli *et al.*, 2004). Consistent with previous publications, we can use the controller's FSM depicted in Fig. 2 as a blueprint to devise the Probabilistic FSM (PFSM or Markov chain) representing the whole swarm at the macroscopic level. The overall PFSM for the system is represented graphically in Fig. 3, using two coupled PFSMs, one representing the robot(s) and one representing the shared turbine environment. At the macroscopic level a state defines the average number of individuals in the same

mode and the same inspection state. The state granularity can be fine tuned in order to achieve an appropriate balance between model complexity and details relevant for the swarm performance metrics (in our case, the time to completion and the energy consumption of the swarm).
All our models are time-discrete (time step T), and characterized by two categories of parameters: state-to-state transition probabilities and behavioral delays. These parameters are computed and calibrated with the same method illustrated by Correll and Martinoli (2006*b*). Values adopted for our models are the same as summarized in (Correll and Martinoli, 2005).

2.1 Model Parameters

In our inspection model, p_e, p_r, and p_w represent the encountering probabilities of blades, robots, and walls respectively. T_e, T_r, and T_w define the *average* time needed for circumnavigating a blade, avoiding a teammate, and avoiding a wall respectively, and T_b is the average time lost on a blade before a beacon is met. While T_r and T_w are calibrated by simple experiments involving only one or two robots, T_e and T_b are functions of blade geometry, the time-out T_{max}, and the signaling range of a robot, respectively, which are detailed in (Correll and Martinoli, 2005).
The probability to hit a beacon p_b is a function of the number of robots acting as a beacon $N_b(k)$ and the number of inspected blades $M_i(k)$

$$p_b(k) = \frac{N_b(k)}{M_i(k)}, \qquad M_i(k) \geq N_b(k) > 0 \tag{1}$$

Finally we calculate the probability to hit a virgin or inspected blade, p_{virgin} and $p_{inspected}$, by multiplying the number of virgin and inspected blades, $M_v(k)$ and $M_i(k)$, with the probability p_e to encounter one blade.
$T_s(k)$ is the time spent signaling as a beacon, and is given by

$$T_s(k+1) = T_s(k) + \tilde{T}_s(k), \tag{2}$$

with $T_s(0) = T_{s,init}$, and $\tilde{T}_s(k) = 0$ in the static case, while $\tilde{T}_s(k) = f(k)$ with $f(k)$ a non-linear function with arbitrary parametrization in the dynamic case, allowing to increase or decrease the time spent as beacon during the experiment.

2.2 Mathematical Description of the Macroscopic Model

From Fig. 3, *right*, we can derive a set of difference equations (DE) to capture the dynamics of the whole system at the macroscopic level. We formulate one DE per considered state and exploit equations stating the conservation of the number of robots and the number of blades to replace one of the DEs.
Given M_0 blades and N_0 robots, the number of robots covering virgin and inspected blades N_v and N_i, the number of robots in obstacle avoidance N_a, the number of robots being a beacon N_b, and the number of robots in search mode N_s are given by (3)–(8) (compare also Fig. 2); the number of virgin blades M_v and the number of inspected blades M_i are calculated by (9)–(10):

$$\begin{aligned}
N_v(k+1) &= N_v(k) + \Delta_v(k) - \Delta_v(k-T_e) && (3)\\
N_i(k+1) &= N_i(k) + \Delta_i(k) - \Delta_b(k) && (4)\\
&\quad -\Delta_i(k-T_e)\Gamma(k-T_e;k) && (5)\\
N_a(k+1) &= N_a(k) + \Delta_r(k) + \Delta_w(k) && (6)\\
&\quad -\Delta_r(k-T_r) - \Delta_w(k-T_w) &&\\
N_b(k+1) &= N_b(k) + \Delta_i(k-T_e)\Gamma(k-T_e;k) &&\\
&\quad +\Delta_v(k-T_e) &&\\
&\quad - \sum_{j=0}^{-\tilde{T}_s(k)} \Delta_v(k-T_e-T_s(k)+j) &&\\
&\quad - \sum_{j=0}^{-\tilde{T}_s(k)} \Delta_i(k-T_e-T_s(k)+j) &&\\
&\quad \cdot\Gamma(k-T_e-T_s(k)+j;k) && (7)\\
N_s(k+1) &= N_0 - N_v(k+1) - N_i(k+1) && (8)\\
&\quad -N_a(k+1) - N_b(k+1) &&
\end{aligned}$$

$$\begin{aligned}
M_v(k+1) &= M_v(k) - \xi_e\Delta_v(k-T_e) && (9)\\
M_i(k+1) &= M_0 - M_v(k+1) && (10)
\end{aligned}$$

with k representing the current time step (and absolute time kT); $k = 0 \ldots n$, n being the total number of iterations (and therefore nT the end time of the experiment). The Δ-functions define the coupling between state variables of the model and can be calculated as follows:

$$\begin{aligned}
\Delta_v(k) &= p_{virgin}(k)N_s(k) && (11)\\
\Delta_i(k) &= p_{inspected}(k)N_s(k) && (12)\\
\Delta_b(k) &= \frac{T}{T_b}p_b(k)N_i(k) && (13)\\
\Delta_r(k) &= p_r(N_0-1)N_s(k) && (14)\\
\Delta_w(k) &= p_w N_s(k) && (15)
\end{aligned}$$

Similar to the collaboration model described by Martinoli *et al.* (2004), the Γ-function represents the fraction of robots that unavailingly waited for collaboration. Here, Γ expresses the fraction of robots that did not encounter a beacon before leaving a blade after T_e.

$$\Gamma(k-T_e;k) = \prod_{j=k-T_e}^{k} \left(1 - \frac{T}{T_b}p_b(k)\right) \tag{16}$$

Due to the possibility of leaving a blade before it has been completely covered due to T_{max}, we introduce a parameter ξ_e in (9), being the percentage of a blade a robot covers on average at each new interaction with it. Similarly to T_e, ξ_e can be calculated from the blade geometry and T_{max} as show by Correll and Martinoli (2005). Note the summation introduced in (7) which is necessary for allowing T_s to be time variant (for $\tilde{T}_s(k) = 0$ the equations simplify to those from Correll and Martinoli (2005)). In short, if T_s is decreased by $\tilde{T}_s$, all robots that became a beacon in the interval $[k - T_s(k) - \tilde{T}_s(k); k - \tilde{T}_s(k)]$ need to continue searching at once. For increasing T_s, no robot shall leave the beacon state for $\tilde{T}_s$. Note, that using this notation, the model can only give valid prediction for $\tilde{T}_s(k)\epsilon] - \infty; 1]$ as for $\tilde{T}_s(k) > 1$, $\tilde{T}_s(k+1)$ will be zero, and thus renders the sum useless.

2.2.1. Initial Conditions The initial conditions are $N_s(0) = N_0$ and $N_a(0) = N_v(0) = N_i(0) = N_b(0) = 0$ for the robotic system (all robots in search mode) while those of the environmental system are $M_v(0) = M_0$ and $M_i(0) = 0$ (all blades virgin). The dynamic parameter $T_s(k)$ is initialized by $T_s(0) = T_{s,init}$. As usual for time-delayed DE, we assume $\Delta_x(k) = N_x(k) = M_x(k) = T_s(k) = 0$ for $k < 0$.

2.3 Swarm Performance Metrics

We consider a composite metric $J = nT + E(k, N_0, N_b)$ for evaluating the performance of the swarm consisting of time to completion nT (terminal cost) and energy consumption (stage cost). For calculating the energy consumption, we use the following model for the individual agent. During every time step T that a robot is moving (Search, Avoidance, or Inspection modes, compare Fig. 2), it consumes η Watt on average, while it consumes γ Watts during every time step it is in beacon mode to account for idling and communication cost. Thus, the swarm energy used from the beginning of the experiment up to time step k can be calculated by

$$E(k, N_0, N_b) = \sum_{j=0}^{k}((N_0 - N_b(k))\eta + N_b(k)\gamma)T, \tag{17}$$

with N_0 the number of robots and $N_b(k)$ the profile of the number of robots acting as a beacon. The task is completed if all blades are inspected ($M_v(n) = \epsilon$), with $0 < \epsilon$ a certain degree of confidence. To compute the time to completion nT, $M_v(n) = 0$ is an easy condition to apply in the experiment. However, in the macroscopic model, this represents a limit condition as $\lim_{k\to\infty} M_v(k) = 0$, and thus we solve the macroscopic model for $M_v(n) = \epsilon$, with ϵ a reasonable small value.

3. OFF-LINE OPTIMIZATION USING MACROSCOPIC MODELS

In this section we formally introduce the dynamic optimization problem. Note that due to its discrete nature the applied metrics are only approximately convex. In particular, due to the assumption that a blade can only be left at its tip, the average time spent on a blade yields discrete values that are a multiple of the time needed to inspect a blade, and thus infinitesimal changes in T_{max} do not necessarily lead to changes in performance. For this reason, we constrained all numerical methods to minimum step sizes of $10s$ for T_{max} and $1s$ for T_s.

3.1 Dynamic Optimization

We are interested in finding an optimal beacon policy, i.e. a profile for $T_s(k)$ that minimizes time to completion nT. We consider T_{max} a time-invariant input, and $\tilde{T}_s(k)$ the time-varying decision variable defining the profile $T_s(k)$. With the dynamics of the system given by (3)–(10) and the same initial conditions (2.2.1), we formulate the optimization problem as follows

$$\underset{T_{max},\tilde{T}_s(0)\ldots\tilde{T}_s(N-1)}{min} J = nT + E(n, N_0, N_b) \tag{18}$$

$$s.t. \quad 0 = M_v(n) - \epsilon$$

In this case, energy consumption at time k (compare (17)) is considered as a stage cost, and $0 = M_v(n) - \epsilon$ is a terminal constraint ensuring all blades have been inspected. For simplifying the dynamic optimization problem we parameterize $\tilde{T}_s(k)$ as follows:

$$\tilde{T}_s(k) = \begin{cases} a \text{ when } k_s \leq k \leq k_s + \Delta k_s, \\ 0 \text{ else.} \end{cases} \tag{19}$$

The profile $T_s(k)$ is hence (see equation (2)) defined by three decision variables $T_{s,init}, k_s$ and Δk_s, and a fixed parameter a (see below). Thus, $T_s(k)$ has the value $T_{s,init}$ until time k_s where it increases until $T_s(k) = T_{s,init} + a\Delta k_s$ at $k_s + \Delta k_s$. The optimization problem can hence be reformulated to

$$\underset{T_{max},T_{s,init},k_s,\Delta k_s}{min} J = nT + E(n, N_0, N_b) \tag{20}$$

$$s.t. \quad 0 = M_v(n) - \epsilon$$

with J the cost and $u = [T_{max}, T_{s,init}, k_s, \Delta k_s]$ the vector containing the decision variables to optimize.

3.2 Extremum-Seeking Control

We use Extremum-Seeking Control based on tracking the necessary condition of optimality (NCO), which is $\frac{\partial J}{\partial u} = 0$, i.e. the first derivative of the cost equals to zero is a necessary condition for optimality. Note, that by constraining minimum step-sizes of our discrete parameters, and exploiting the fact that the macroscopic model is predicting the *average* performance of the system, we can assume J to be convex, differentiable, and $\frac{\partial^2 J}{\partial^2 u}$ to be positive definite. NCO allows treating the optimization problem as a control problem, which we formulate as follows

$$\dot{u} = -\alpha \frac{\partial J}{\partial u}, \tag{21}$$

where α is the gain of the integral controller that drives $\frac{\partial J}{\partial u}$ to zero[1].
As $\frac{\partial J}{\partial u}$ cannot be calculated analytically in this case, we add a perturbation in form of a sinusoidal function to individual components of the vector u. The period length of the perturbation is hereby chosen to be larger than the time needed to reach steady-state. Note that the NCO tracking control loop is time-continuous, and is independent of the type (continuous-time or discrete-time) of the plant to optimize. Hence, we estimate $\frac{\partial J}{\partial u}$ as follows

$$\frac{\partial J}{\partial u} \approx \frac{J(u + p(j)) - J(u)}{p(j)}, \tag{22}$$

with $p(j)$ the perturbation vector at run j.

4. RESULTS

All DEs were solved until the number of virgin blades were reduced to $\epsilon = 0.1$. If not stated otherwise, all scenarios consider 20 robots and 16 blades, the time discretization of the system is $T = 1s$, and energy consumption $\eta = 1W$ and $\gamma = 1W$.

4.1 Static Optimization

In a first step, we validated findings from Correll and Martinoli (2006*b*) and Correll and Martinoli (2005) for constant $T_s(k) = 0$ using the MATLAB® function *fmincon* that finds a minimum of an unconstrained multivariable function by Sequential Quadratic Programming. We found $T^*_{max} \approx 20s$, yielding an optimal inspection time of $n^*T = 144s$ and energy consumption $E^* = 2900J$. Introducing different constraints on the total energy consumption, yields more interesting results. For instance for arbitrary chosen $E(nT) <= 2720J$, the optimization routine converges to $T_s = 5s$ leading to $n^*T = 149s$. For $E(n^*) <= 2500J$ instead, we found $T^*_s = 49s$, yielding $n^*T = 219s$ and $E^* = 2705J$, i.e. the constraints could not be satisfied.

[1] One can show using Lyapunov theory that the optimal feedback solution for convex cost functions is always stable

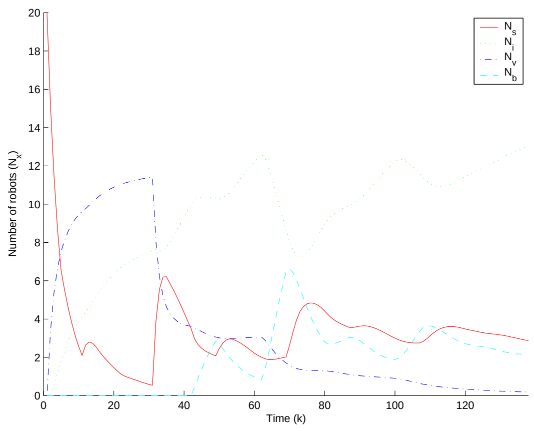

Fig. 4. Development of the state variables with an optimal policy for $T_s(k)$ ($k^*_s = 40$).

4.2 Dynamic Optimization

For finding an optimal beacon policy, we constrained the parameter a (compare Equation 19) to take discrete values in the interval $[-1; +1]$, where $a = 0$ is the static case treated above. In other words, we consider two policies, first to relax a beacon policy by decreasing T_s over time, and second, to foster a beacon policy by increasing T_s. We performed optimization for $8 \leq N_0 \leq 40$ with $M_0 = 16$. We observe a bifurcation of the system for the ratio of robots to blades, as an optimal profile for $T_s(k)$ could only be found for $M_0 \leq N_0$. Optimization using *fmincon* yields $T^*_{s,init} = 0$ and $\Delta k^*_s = 0$ for $a = -1$, i.e. starting with a beacon policy always decreases performance.
On the other hand, setting $a = +1$, yields an optimal policy $T^*_{s,init} = 0$, $k^*_s = 40$, $\Delta k^*_s = 7$ leading to an inspection time of $n^*T = 138s$ using only $E^* = 2494J$, for $N_0 = 20$ robots. The evolution of the state variables for this particular case is depicted in Figure 4.

Using the extremum-seeking control scheme instead of *fmincon*, with $\dot{u}$ according to (21), and a sinusoidal perturbation (values for $p(j)$ from -4 to 4 following a sinusoidal pattern), leads to an optimum $n^*T = 138s$ and $E^* = 2524J$ for $T^*_{s,init} = 0$, $k_sT^* = 40$, and $\Delta k^*_s = 6$.

5. DISCUSSION

Imposing a constraint on the energy ($E(n^*) <= 2720J$) yielded an optimal signaling time of $T^*_s = 5s$. This value intuitively makes sense, as the maximum time a beacon policy can save is the time to fully surround a blade ($T_b = 20s$), but only if the beacon is indeed met by another robot. Longer signaling times slows down inspection progress as the involved robots can not be used for search, which in turn leads to higher overall energy consumption. Trying to find an optimal policy for further minimizing energy consumption ($E(n^*) <= 2500J$) failed, and suggests a minimal

energy consumption of $E = 2705J$ in $n^*T = 219s$ for $T_s^* = 49s$ (static optimization).
Using dynamic optimization, we discover an optimal profile for $T_s(k)$ resulting in a dynamic beacon policy. Unfortunately, the optimal time to employ the beacon policy, k_s^* is a function of the number of robots — the more robots there are, the earlier inspection finishes, and hence the smaller will be k_s^*—as well as the ratio of robots to blades. Nevertheless, it might be possible for the robots, even in a real scenario, to estimate the necessary parameters online, for instance by measuring the ratio of encountered blades to encountered robots. Extremum-seeking control allowed to reach the optimum after a few iterations (between 5 and 25), each involving the estimation of the gradients for the four decision variables from measurements of the system.

6. CONCLUSION AND FURTHER WORK

Optimal Control theory has shown to provide powerful tools for optimizing self-organized robotic swarms in an inspection task, and provided non-obvious control policies that potentially increase performance when applied to a real system. In particular, our optimal control of the beacon-on time is directly affecting the nonlinear mechanism at the core of the self-organized coordination mechanism. Here, NCO tracking methods are attractive since they allow treating the optimization problem as a control problem with all the advantages related to sensitivity reduction and disturbance rejection in an analytical tractable fashion.
It is also to note that the performance gain for the optimal beacon policy in this case study is very small (around 5% compared with a system without collaboration) and thus requires large number of experiments in order to show statistically significant results (in the order of 50-100 runs), especially as the standard deviation of the performance metric is large due to the probabilistic nature of self-organization. Finally, as model parameters like interaction probability and interaction time are uncertain as they are computed using a simple heuristic or calibrated by simple experiments involving only a few robots, the optimum of the real system might be different from the optimum observed on the model. At this point it would be interesting to extend the presented extremum seeking control scheme for parameter estimation (involving the real system in the loop), and hence complement our current methodology for modeling swarm robotic systems.

Acknowledgments

The authors would like Bala Srinivasan, École Polytechnique Montréal, for helpful discussion on optimal control. Both authors are currently sponsored by a Swiss NSF grant (contract Nr. PP002-68647/1).

REFERENCES

Bonabeau, E., M. Dorigo and G. Theraulaz (1999). *Swarm Intelligence: From Natural to Artificial Systems.* SFI Studies in the Science of Complexity, Oxford University Press, New York, NY, USA.

Correll, N. and A. Martinoli (2005). Modeling and analysis of beacon-based and beaconless policies for a swarm-intelligent inspection system. In: *Proceedings of the International Conference on Robotics and Automation (ICRA).* pp. 2488–2493.

Correll, N. and A. Martinoli (2006*a*). Collective inspection of regular structures using a swarm of miniature robots. In: *Proceedings of the 9th International Symposium of Experimental Robotics (ISER), 2004.* Springer Tracts for Advanced Robotics (STAR), Vol. 21. pp. 375–385.

Correll, N. and A. Martinoli (2006*b*). Modeling and optimization of a swarm-intelligent inspection system. In: *Proceedings of the 7th Symposium on Distributed Autonomous Robotic System (DARS), 2004.* Springer Distributed Autonomous Systems VI. pp. 369–378.

Cortés, J., S. Martínez, T. Karatas and F. Bullo (2004). Coverage control for mobile sensing networks. *IEEE Transaction on Robotics and Autonomous Systems* **20**(2), 243–255.

Jadbabaie, A., J. Lin and A. S. Morse (2003). Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control* **48**(6), 988–1001.

Kirk, D. E. (2004). *Optimal Control Theory: An Introduction.* Dover Publications.

Martinoli, A., K. Easton and W. Agassounon (2004). Modeling of swarm robotic systems: A case study in collaborative distributed manipulation. *Int. Journal of Robotics Research* **23**(4), 415–436.

Milutinovic, D., P. Lima and M. Athans (2003). Biologically inspired stochastic hybrid control of multi-robot systems. In: *Proceedings of the 11th International Conference on Advanced Robotics (ICAR).*