**2008 IEEE International Conference on Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008**

# Communication-Aware Trajectory Tracking

Magnus Lindhé and Karl Henrik Johansson
ACCESS Linnaeus Centre
School of Electrical Engineering
Royal Institute of Technology
SE-100 44 Stockholm, Sweden
{lindhe|kallej}@ee.kth.se

*Abstract*— **This paper investigates the scenario of a robot making a tradeoff between tracking a time-varying reference trajectory and stopping to communicate at points where the radio signal strength is high. Under the assumption that the signal is subject to multipath fading, we formulate this as a hybrid optimal control problem with penalties on tracking error, communication buffer length and control power. The problem is then solved using relaxed dynamic programming, resulting in control laws for the discrete switching sequence and the continuous control. We finally illustrate the results through simulations under non-ideal conditions, confirming that the system maintains a bounded buffer size and zero-mean tracking error.**

## I. INTRODUCTION

In many robotic applications, such as surveillance, environmental sampling or mapping, each robot performs some desired motion while generating sensor information that must be sent to a base or other robots. Sensing at the wrong place is as bad as getting relevant measurements that cannot be transmitted, so the overall performance of the system depends on making a proper tradeoff between acquiring data and ensuring good communication.

Such tradeoffs have been investigated for collaborative sensing [1], [2], where the agents find positions that give good sensor resolution *and* good inter-agent communication. Esposito *et al.* [3] have shown how to move a formation among obstacles under a line of sight communication constraint and Arkin *et al.* [4] have studied how to search an indoor environment while ensuring that all searchers stay connected with an outside base. These papers all use simple communication models; the signal quality depends on the distance or on a line of sight condition. Recently we proposed a communication model that in many cases is more suitable: multipath fading [5]. This effect arises if several reflections of the same signal reach the receiver. The signal strength will then fluctuate due to positive or negative interference between the reflections, and it varies over distances in the order of a wavelength (e.g., about 12 cm for a 2.4 GHz signal). We showed by measurements that multipath fading can be a significant effect indoors and presented a method

This work was partially supported by the Swedish Defence Materiel Administration (FMV) through the TAIS programme 297316-LB704859, the European Commission through the RUNES and HYCON projects, the Swedish Research Council and the Swedish Foundation for Strategic Research.

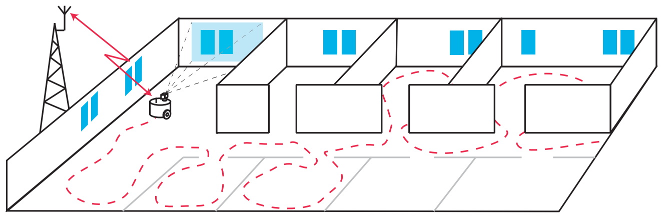

Fig. 1. Illustration of the considered scenario: A robot follows a patrol trajectory in an office building, sending infrared camera imagery of each room to a base station. Since the signal strength fluctuates due to multipath fading, the robot has to make a tradeoff between stopping at good points to communicate and keeping up with the reference trajectory.

to find good positions for robots performing stationary tasks so that the fading could be exploited.

The main contribution of this paper is to study dynamic communication-aware robot positioning: If a robot tracks a time-varying reference trajectory in a multipath fading environment, how should the tradeoff between tracking and communication be done? An example scenario is illustrated in Fig. 1: During night, a robot is patrolling one floor of an office building, collecting infrared camera imagery of each room that is transmitted to an operator on the ground floor. The robot follows a predetermined trajectory, but can adjust its movement along the trajectory to ensure that the images are fed to the operator with minimum delay. Due to multipath fading, the link capacity from the robot to a base station varies when it drives, giving a certain average capacity. But the robot can also choose to stop at a point with stronger signal, thereby emptying its buffer faster, at the expense of lagging behind the reference and having to use more power to catch up. We ask: What is the optimal way to switch between driving and stopping, and how should the robot accelerate, to minimize the tracking error, transmission buffer length and power consumption in its motors? This problem lends itself well to a hybrid optimal control formulation. It is well known that solving such problems can be problematic due to the "curse of dimensionality". Therefore we have applied the method of *relaxed dynamic programming*, proposed by Lincoln and Rantzer [6], that alleviates this issue by approximating the value function with small relative error.

This paper is organized as follows: In the next section we formulate the problem. In Section III, we provide a model for the robot dynamics. We then model the onboard buffer and the communication link for an environment with multipath fading. In Section IV, we derive the system dynamics as a switched linear system and formulate an infinite-horizon hybrid optimal control problem. Using relaxed dynamic programming, we present an algorithm to compute a controller for both the switch sequence and the continuous input. In Section V, we simulate the resulting closed-loop system to illustrate its properties under different non-ideal conditions. Finally we conclude in Section VI, and indicate some possible directions of future research.

## II. PROBLEM FORMULATION

We consider a robot with state $x$, input $u$, position $p(x)$ and dynamics $\dot{x} = f(x,u)$, supposed to follow a time-varying reference trajectory $p_{ref}(t)$. The robot collects information at a rate $r$ and tries to transmit it to a base station over a multipath fading channel. Data are stored in a buffer of length $z \geq 0$ and we assume that the environment is static, which means that the fading is a function of the state of the robot. The buffer dynamics can thus be expressed as $\dot{z} = g(r,x)$.

We can then formulate our problem: *Find a control law $u = u(x)$ such that $z$ is small and $p(t)$ is close to $p_{ref}(t)$.* The control objective is thus to keep the communication latency small while tracking the desired trajectory.

## III. ROBOT AND COMMUNICATION MODELS

In this section, we define a model for the robot and reduce it to one-dimensional movement along the reference trajectory. It is formulated as a hybrid model, switching between driving and standing still. We then present a model for the communication buffer and how the radio link capacity varies with the state of the robot.

### A. Robot Model

Our approach can be used for several types of robots, such as single or double integrators or differential drive. Here we consider a differential drive robot with position $p \in \mathbb{R}^2$, bearing $\theta \in [0, 2\pi[$ and controls $(v, \omega)$. The dynamics are:

$$\begin{cases} \dfrac{\mathrm{d}p_1}{\mathrm{d}t} = v(t)\ \cos\theta(t) \\ \dfrac{\mathrm{d}p_2}{\mathrm{d}t} = v(t)\ \sin\theta(t) \\ \dfrac{\mathrm{d}\theta}{\mathrm{d}t} = \omega(t). \end{cases}$$

We assume that the robot has a given reference trajectory $p_{\mathrm{ref}}(t)$ and is capable of following it, i.e., there exist $v_{\mathrm{ref}}(t)$ and $\omega_{\mathrm{ref}}(t)$ so that $p(t) = p_{\mathrm{ref}}(t)$. To vary the velocity along the trajectory, we introduce the virtual time $s(t)$, defined by

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \gamma(t) > 0,\ s(0) = 0.$$

Replacing the time $t$ with $s(t)$ allows us to speed up the system or slow it down, by applying the controls

$$v(s(t)) = \gamma(t) v_{\mathrm{ref}}(s(t))$$
$$\omega(s(t)) = \gamma(t) \omega_{\mathrm{ref}}(s(t)).$$

We consider the one-dimensional movement of the robot along the reference trajectory and define the position of the robot, relative to the reference, as

$$\Delta(t) = \int_0^t [v(t) - v_{\mathrm{ref}}(s(t))]\ \mathrm{d}t.$$

We also introduce the relative velocity $\varphi(t) = \frac{\mathrm{d}\Delta}{\mathrm{d}t}$. To achieve a relative velocity $\varphi(t)$ we set the scaling

$$\gamma(t) = 1 + \frac{\varphi(t)}{v_{\mathrm{ref}}(s(t))}. \tag{1}$$

To get smooth motion, we use the acceleration $u(t)$ as control. We further want to model the fact that many kinds of robots only consume negligible power when breaking, using disc breaks or by short-circuiting its electric motors. So we consider the robot to have two discrete modes: `drive` and `stop`. In the `stop` mode, we let the relative velocity be self-stabilizing to the value of $-v_{\mathrm{ref}}$. The motion along the reference trajectory is described as:

$$\texttt{stop:} \begin{cases} \dot{\Delta} = \varphi \\ \dot{\varphi} = -k_v(\varphi - v_{\mathrm{ref}}) \end{cases} \quad \texttt{drive:} \begin{cases} \dot{\Delta} = \varphi \\ \dot{\varphi} = u \end{cases}.$$

The parameter $k_v \gg 1$ is chosen to ensure fast convergence of $\varphi$ to $-v_{\mathrm{ref}}$ when stopping.

### B. Data Buffer Model

The data buffer onboard the robot has size $z$, inflow rate $r$ and outflow (or link capacity) $c$, so its dynamics are

$$\dot{z} = r - c.$$

Next we describe how $c$ varies with the mode of the robot. Note that the buffer is lossless, which means that no packets are discarded.

### C. Communication Model under Multipath Fading

The attenuation (or gain) due to multipath fading varies over robot movements of fractions of a wavelength. In Fig. 2, we illustrate a representative dataset from measurements. The graph shows the received signal strength (RSS) for a robot as it moves along a straight line in a lab room with computers and equipment reflecting the incoming signal. For details, see [5]. The average RSS of −58 dBm is marked in the figure, but almost everywhere, the robot is only a few cm away from a peak at the higher level of −54 dBm, which is also marked. A gain of 4 dB may seem small, but can give a substantial link capacity increase. To illustrate the increase, we use a result by Zuniga and Krishnamachari [7]. They derive an expression for the packet reception rate (PRR) as function of the signal-to-noise ratio (SNR) for the MICA2 sensor motes [8]. The motes have a data rate of 19.2 kbit/s and a PRR of

$$\mathrm{PRR} = \left(1 - \frac{1}{2} e^{-\frac{\mathrm{SNR}}{1.28}}\right)^{8F},$$

where $F$ is the frame size, i.e., the number of bytes in each packet. In Fig. 3, we have illustrated what the resulting link capacity would be for $F = 50$. A gain in RSS gives the same

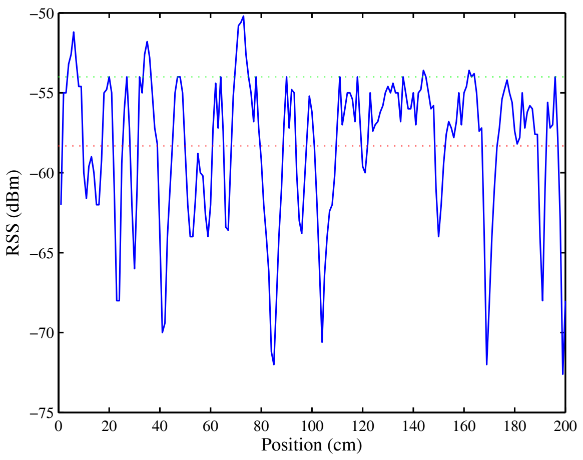

Fig. 2. Measurement results from a lab room, where the received signal strength (RSS) varies due to multipath fading. The lower dashed line denotes the average level, while the upper line is chosen by hand, so that a point with that RSS can be found within a few cm from almost any position.

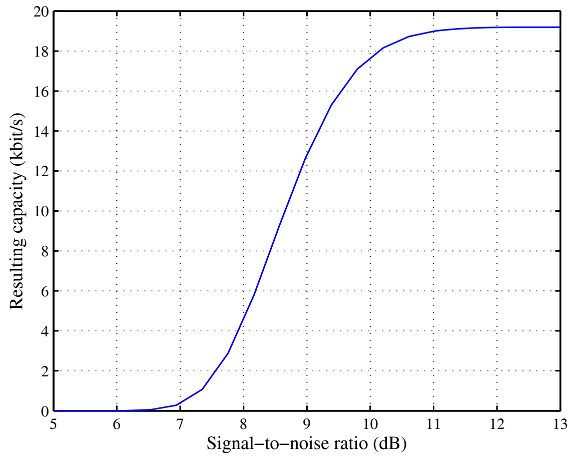

Fig. 3. An illustration of the link capacity for a MICA2 sensor mote as a function of the signal-to-noise ratio. When the link is on the limit of losing contact, gaining just a few dB can have a large impact on bandwidth.

gain in SNR, so for a robot on the limit of losing contact, gaining 4 dB could mean more than a tenfold increase in bandwidth.

Motivated by this, we derive a simple model for the communication channel: In the `drive` state, the radio hardware smoothes the RSS variations, producing an average buffer outflow $c_d$. But when the robot is in the `stop` state, we assume that it can instantly find a point with a higher signal strength and a higher capacity $c_s$. It is important to point out that this approach is most useful in the interval $c_d < r < c_s$, since if the inflow is lower than $c_d$ the robot is never forced to stop, and if it is larger than $c_s$, some higher-level protocol must discard data to stop the buffer from overflowing. Also note that this does not require accurate navigation, since the robot can always find a high-capacity position by just driving a few centimeters in any direction.

## IV. HYBRID OPTIMAL CONTROL SOLUTION

In this section, we formulate the problem of this paper as a hybrid optimal control problem. We then present relaxed dynamic programming as a way of finding an approximate solution. We state an algorithm that computes a value function from which we can derive a control law. Finally we show how to do this computation in an efficient way.

### A. Switched Linear System

To describe the whole system, we collect the robot and buffer states in the same state vector. We include an integral state $\Delta_I$ to allow the controller to attenuate a static error in $\Delta$. We finally add a constant element to the state vector, which allows writing the system on linear form. This yields

$$\dot{x} = A_\sigma x + B_\sigma u,\ x = (\Delta, \varphi, \Delta_I, z, 1)^T,$$

where the controls are $u \in \mathbb{R}$ and $\sigma \in \{0,1\}$, which correspond to `stop` and `drive`, respectively, and

$$A_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & -k_v & 0 & 0 & -k_v v_{\text{ref}} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r - c_s \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},\ B_0 = 0$$

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r - c_d \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},\ B_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

We consider a sampled version of the continuous-time system above, where we suppose $u$ is kept constant between sampling times and $\sigma$ only switches at sampling instances. With sampling time $\tau$, we can express the discrete dynamics as

$$x[n+1] = f(x[n], u[n], \sigma[n]) = \Phi_{\sigma[n]} x[n] + \Gamma_{\sigma[n]} u[n],$$

where

$$\Phi_\sigma = e^{A_\sigma \tau} \text{ and } \Gamma_\sigma = \int_0^\tau e^{A_\sigma s} B_\sigma\, ds.$$

### B. Cost Function

To maintain low latency and margin for unexpected buffer inflow, it is desirable to keep the buffer size low. At the same time we also want to stay close to the reference trajectory and limit the control magnitude. Both $\Delta$ and $z$ will in general not simultaneously converge to zero, so we introduce a decay factor $\lambda^n$, with $\lambda < 1$, to get a finite cost in an infinite horizon control problem. Now, given an initial condition $x_0$, the optimal control problem is defined as

$$\begin{aligned} \min_{\sigma[n], u[n]} \quad & \sum_{n=0}^{\infty} \left( x^T[n] Q x[n] + R u^2[n] \right) \lambda^n \qquad (2) \\ \text{s.t.} \quad & x[n+1] = f(x[n], u[n], \sigma[n]) \\ & x[0] = x_0 \\ & x_4 \geq 0, \end{aligned}$$

where $Q = Q^T$ is positive semidefinite and $R$ is a positive constant.

## C. Dynamic Programming

Dynamic programming is based on approximating the optimal value function (also called cost-to-go) at state $x[m]$, defined as

$$V^*(x[m]) = \min_{\sigma[n],u[n]} \sum_{n=m}^{\infty} \ell(x[n],u[n])\lambda^n,$$

where

$$\ell(x,u) = x^T Qx + Ru^2.$$

Once we have $V^*(x)$, we can derive the optimal control law as

$$(u^*(x),\sigma^*(x)) = \operatorname{argmin}_{u,\sigma}\left\{\lambda V^*(f(x,u,\sigma)) + \ell(x,u)\right\}.$$

We introduce an approximate value function $V_k(x): \mathbb{R}^5 \to \mathbb{R}$ such that $\lim_{k\to\infty} V_k(x) = V^*(x)$ and use value iteration to recursively compute it:

$$V_{k+1}(x) = \min_{u,\sigma}\left\{\lambda V_k(f(x,u,\sigma)) + \ell(x,u)\right\},\ V_0 = 0. \tag{3}$$

For a given $k$, the iterate $V_k(x)$ answers the question "what is the lowest possible cost for $k$ time steps of the system trajectory, given that it starts in $x$?" The problem is that, if applied naively, value iteration requires that we consider *all* possible switching sequences of length $k$ steps, so the complexity of our hybrid optimal control problem grows exponentially with the horizon length $k$. This "curse of dimensionality" is a well known drawback of dynamic programming. Before we present a way to avoid this, we will see how the optimal control $u^*$ can be computed for a known switching sequence of length $k$.

To facilitate the notation, let $\Phi = \Phi_{\sigma[n]}$ and $\Gamma = \Gamma_{\sigma[n]}$ for some given mode $\sigma[n]$. We also assume that, at time $n+1$, the value function can be written on quadratic form $V^*(x[n+1]) = x^T[n+1]P[n+1]x[n+1]$, where $P[n+1]$ is a symmetric positive semidefinite matrix. Then the optimal cost at time $n$ is $V^*(x[n]) = x^T[n]P[n]x[n]$, where

$$\begin{aligned} P[n] &= \lambda\Phi^T P[n+1]\Phi + Q - \lambda\Phi^T P[n+1]\Gamma \\ &\quad\times\left[\lambda\Gamma^T P[n+1]\Gamma + R\right]^{-1}\Gamma^T P[n+1]\Phi\lambda \end{aligned} \tag{4}$$

and $P[n]$ is positive semidefinite [9]. Further, the optimal control is

$$u^*(x[n]) = -\left[R + \lambda\Gamma^T P[n+1]\Gamma\right]^{-1}\lambda\Gamma^T P[n+1]\Phi\, x[n]. \tag{5}$$

If we initialize $P[k] = 0$, for a known switching sequence of length $k$ we can thus iteratively compute the optimal cost $V^*(x[0])$ and control signal $u^*[n], n \in \{0,\dots,k-1\}$.

## D. Relaxed Dynamic Programming

We define a discrete mode trajectory of length $k$ as $\sigma_j^k : \{0,1,\dots,k\} \to \{0,1\}$. Further, let $N_k$ be the number of such trajectories in the set $\{\sigma_1^k,\dots,\sigma_{N_k}^k\}$ of candidates for optimality. Now let $\Pi_k = \{P_1^k,\dots,P_{N_k}^k\}$ be the set of matrices $P_j^k$ such that the cost associated with $\sigma_j^k$ is $V_k(x[0]) = x^T[0]P_j^k x[0]$. Using a sufficiently rich set $\Pi_k$, we can now parameterize the approximate value function in a way that can be used to perform the iteration (3):

$$V_k(x) = \min_{j\in\{1,\dots,N_k\}} x^T P_j^k x.$$

As mentioned above, even for small $k$, the set $\Pi_k$ becomes prohibitively large if we do not discard some candidate switching sequences during the recursion. Lincoln and Rantzer's [6] method of relaxed dynamic programming does just that: at each iteration, it retains only the candidates $P_j^k$ that are needed to represent the value function with a given bounded relative error. If this bound is sufficiently large, the number of candidates will converge to a finite value as $k \to \infty$.

More formally, the idea is to find an approximation $V_k(x)$ of the optimal value function such that, for $\overline{\alpha} \geq 1$ and $\underline{\alpha} \leq 1$,

$$\begin{aligned} &\min_{u,\sigma}\{\lambda V_k(f(x,u,\sigma)) + \underline{\alpha}\ell(x,u)\} \leq V_k(x) \\ &\quad\leq \min_{u,\sigma}\{\lambda V_k(f(x,u,\sigma)) + \overline{\alpha}\ell(x,u)\}\ \forall\ x. \end{aligned} \tag{6}$$

Using the appropriate "slack", the cost-to-go function can be parameterized by a much smaller set $\Pi_k$, and we can discard many candidate switching sequences at each iteration step. For the discarding procedure, we define $\underline{\Pi_k} = \{\underline{P}_1^k,\dots,\underline{P}_{N_k}^k\}$ as the set of matrices $\underline{P}_j^k$ such that $\underline{\alpha}$ times the cost for the switching sequence $\sigma_j^k$ is $x[0]^T\underline{P}_j^k x[0]$. The set $\overline{\Pi}_k$ and the matrices $\overline{P}_j^k$ are defined analogously, using $\overline{\alpha}$. The method to find $V_k(x)$ is presented in Algorithm 1.

**Algorithm 1** Relaxed Dynamic Programming

```
1: $k := 0$, $\Pi_0 = 0_{n\times n}$
2: while (6) is not fulfilled do
3:     $k := k+1$
4:     Form $\overline{\Pi}_k$ and $\underline{\Pi_k}$ by propagating the matrices
       in $\Pi_{k-1}$ one step backwards in time, both with
       $\sigma = 0$ and $\sigma = 1$, as defined in (4). This yields
       $N_k = 2\ \mathrm{card}(\Pi_{k-1})$.
5:     Sort the sets $\overline{\Pi}_k$ and $\underline{\Pi_k}$ so that $\mathrm{tr}\overline{P}_1^k \leq \ldots \leq \mathrm{tr}\overline{P}_{N_k}^k$ and
       $\overline{P}_j^k \geq \underline{P}_j^k\ \forall\ j$.
6:     $\Pi_k := \emptyset$, $i := 1$
7:     while $i \leq N_k$ do
8:         if $\nexists$ a convex combination $P$ of matrices in $\Pi_k$ such
           that $P \leq \overline{P}_i^k$ then
9:             Add $\underline{P_i^k}$ to $\Pi_k$.
10:        end if
11:        $i := i+1$, $N_k := \mathrm{card}(\Pi_k)$
12:    end while
13: end while
```

Note that step 8 of the algorithm is an S-procedure test to see if there exists an $x$ such that

$$x^T\overline{P}_i^k x < \min_{P\in\Pi_k} x^T Px.$$

If not, then $P_i^k$ is not needed to represent the value function with sufficient accuracy. Also note that by ordering the matrices by trace, we ensure that smaller matrices are added

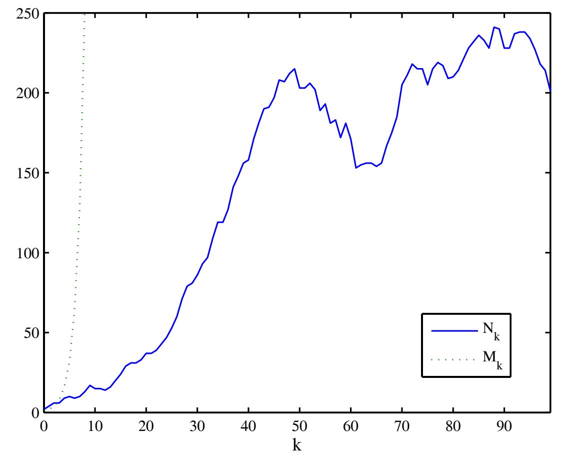

Fig. 4. The number $N_k$ of matrices in $\Pi_k$ needed to represent the value function at each iteration step. The vertical axis shows the number of candidate matrices $N_k$ and $M_k$ as a function of the iteration step $k$. $N_k$ stops increasing, indicating that (6) is fulfilled, after about 50 iterations. Without discarding any candidates, the complexity would grow as $M_k = 2^k$, which is illustrated for comparison.

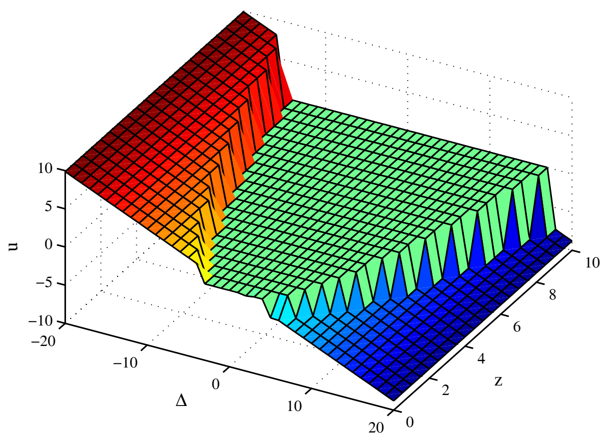

Fig. 5. The resulting control law for the subset $\varphi = -v_{\text{ref}}$, $\Delta_I = 0$ (corresponding to standing still with an empty integral state in the controller). To also illustrate $\sigma(x)$, we have forced the control to $u = 0$ where $\sigma(x) = 0$. As one would expect, there is a `stop` region for small $\Delta$. If $\Delta$ decreases, the controller accelerates the robot and if $\Delta$ becomes too large, it slows the robot down.

first to $\Pi_k$, which in practice means that we will add fewer elements.

The stopping criterion for the iteration means that no more candidate switching sequences need to be added to represent the value function. This happens when the number $N_k$ of candidates stops growing, and to determine this we have used the graph depicted in Fig. 4. For comparison we also included the number of candidates $M_k$ that would have to be considered using normal dynamic programming. Note that $N_k$ does not converge, but rather stops growing and then displays random variations due to numerical effects and small random perturbations in the sorting of $\overline{\Pi}_k$ to make the search more efficient. The figure shows the result for $\overline{\alpha} = \underline{\alpha}^{-1} = 2$ and $\lambda = 0.9$, which are the parameters used to compute the controller used in all simulations.

## E. Resulting Controller

Using the approximation of the value function, we could find the optimal mode $\sigma^*(x[n])$ as the first mode in the switch sequence corresponding to the matrix

$$P^* = \operatorname{argmin}_{P \in \Pi_{100}} x[n]^T P x[n].$$

The optimal control $u^*(x[n])$ was then computed using (5), substituting $P^*$ for $P[n+1]$. In a resource-constrained robot, this could also be precomputed and stored as a look-up table of feedback gains $L_j$, each associated with a switching sequence $\sigma_j^{100}$. The control signal would then be $u^*(x[n]) = -L_j x[n]$. The resulting control law is plotted in Fig. 5, for the subset $\varphi = -v_{\text{ref}}$, $\Delta_I = 0$ of the state space.

## V. SIMULATION RESULTS

In this section, we first present an illustration of a system trajectory using the previously derived controller. We then investigate the sensitivity of the closed-loop system to disturbances in buffer size and link capacity. In all simulations, the sampling time is $\tau = 0.1$ s. We set $k_v = 100$, $R = 1$ and $Q = \text{diag}(1, 0, 1, 5, 0)$. We thus penalize the position error and integrated error $\Delta$ and $\Delta_I$, respectively. To reduce the static error in $z$, we use a higher penalty $Q_{44}$.

## A. Following a Curved Path

Fig. 6 illustrates the system following a curved reference trajectory corresponding to $v_{\text{ref}} \equiv 1$ and $\omega_{\text{ref}} = \sin(0.8t)$. This means that the reference is moving at constant velocity along the path, while the robot varies $\gamma$ as in (1) to perform the communication-aware tracking. The figure consists of periodical samples of the robot state, with the height of the ball over the robot indicating buffer size. Thus the robot stops at some points to empty its buffer. We have used the exaggerated rates $r - c_s = 1$ and $r - c_d = -1$ to illustrate the behavior of the system more clearly.

## B. Limit Cycle and Buffer Disturbance Rejection

We have then simulated the system starting with an empty buffer and perfect reference tracking. As seen in Fig. 7, it approaches a limit cycle with a period time of 5.6 s, where is spends 50% of the time in each mode. The relative position $\Delta$ oscillates around zero while there is still a small static error in $z$. However, at $t = 40$ s, extra data is added to the buffer, and this impulse disturbance is successfully attenuated. Here we also used $r - c_s = 1$ and $r - c_d = -1$.

## C. Robustness to Capacity Variations

As indicated in the derivation of the communication model, the actual link capacity $c_s$ at the position where the robot stops can vary from the predicted value. We have tested the robustness of the closed-loop system to this model error by adding zero-mean white gaussian noise with standard deviation 2 to $c_s$. With $c_s = 3$, $c_d = 1$ and $r = 2$, the

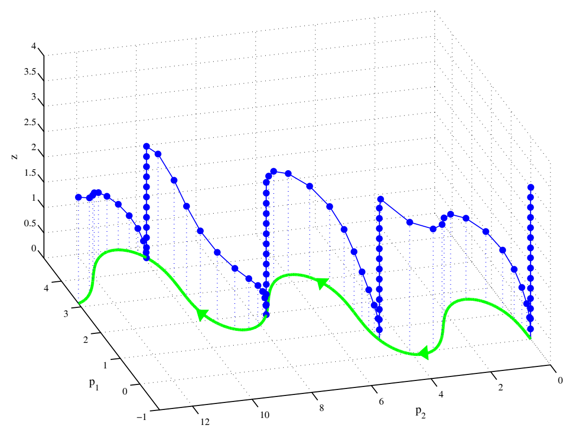

Fig. 6. A trajectory of the system in the $(p_1, p_2, z)$-space, sampled with regular intervals. The robot follows the reference trajectory (thick green line) while stopping from time to time to reduce the buffer size. The robot motion is from right to left.

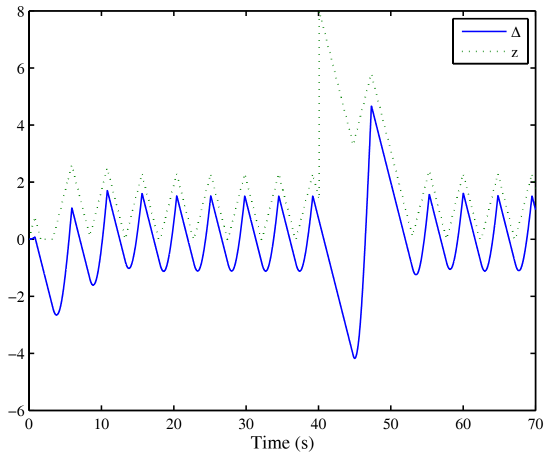

Fig. 7. An example of a trajectory for the system, starting with an empty buffer and with perfect reference tracking. The solid blue line represents the relative position $\Delta$ and the dashed green line is the buffer size $z$. The robot approaches a limit cycle with a period time of 5.6 s. One can see that the integral state of the controller forces $\Delta$ to oscillate around zero, while there is still a static error in the buffer size.

simulations indicate that the system still oscillates around $\Delta = 0$ and a maintains a bounded buffer size $z$.

## VI. CONCLUSIONS

We have studied a robot performing communication-aware trajectory tracking in a multipath fading environment. The problem can be cast as a hybrid optimal control problem, which can in turn be solved with sufficient accuracy using relaxed dynamic programming. The resulting controller can be stored in look-up tables and thus used also on resource-constrained robots. The only prior information needed about the radio link is the capacities $c_s$ and $c_d$, no map of the signal strength is needed. The closed-loop system was simulated under various conditions and it maintains a bounded buffer size and zero-mean tracking error.

The radio model is derived under the assumption of stationary multipath fading, but this approach could be used whenever the signal strength varies in space and it is simple to find good positions to stop at. Examples of this could be an underwater robot that can surface to communicate, or a robot searching office rooms, where the signal may be stronger in the corridor or near windows.

In our ongoing work, we plan to implement the proposed control law on a physical robot, validating our approach with real-world radio characteristics. An interesting future research question is how to control the robot if it is not restricted to stay on the reference path, but has the option of extending its trajectory to visit more potential communication positions. Especially for slowly moving reference trajectories, this could increase the probability of finding a spot offering higher link capacity, at the expense of reference tracking error.

## ACKNOWLEDGEMENTS

The authors would like to thank Henrik Sandberg for helpful discussions and the authors of [6] for kindly providing numerical routines for relaxed dynamic programming.

## REFERENCES

[1] T. Chung, J. Burdick, and R. Murray, "A decentralized motion coordination strategy for dynamic target tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[2] D. O. Popa and C. Helm, "Robotic deployment of sensor networks using potential fields," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2004.

[3] J. M. Esposito and T. W. Dunbar, "Maintaining wireless connectivity constraints for swarms in the presence of obstacles," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2006.

[4] R. Arkin and T. Balch, "Line-of-sight constrained exploration for reactive multiagent robotic teams," in *Proceedings of 7th International Workshop on Advanced Motion Control*, 2002.

[5] M. Lindhé, K. H. Johansson, and A. Bicchi, "An experimental study of exploiting multipath fading for robot communications," in *Proceedings of Robotics: Science and Systems*, 2007.

[6] B. Lincoln and A. Rantzer, "Relaxing dynamic programming," *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1249–1260, 2006.

[7] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *Proceedings of the First IEEE International Conference on Sensor and Ad hoc Communications and Networks*, 2004.

[8] Crossbow Corporation, "Mica2 datasheet," document part number 6020-0042-08 Rev A. Last visited 2007-10-29. [Online]. Available: http://www.xbow.com/Products/productdetails.aspx?sid=174

[9] K. J. Åström and B. Wittenmark, *Computer-controlled systems: theory and design*. Prentice-Hall, 1997.