Scheduled Perception for Energy-Efficient Path Following

Scheduled Perception for Energy-Efficient Path Following

Peter Ondrúška, Corina Gurău, Letizia Marchegiani, Chi Hay Tong and Ingmar Posner

***Abstract*— This paper explores the idea of reducing a robot's energy consumption while following a trajectory by turning off the main localisation subsystem and switching to a lower-powered, less accurate odometry source at appropriate times. This applies to scenarios where the robot is permitted to deviate from the original trajectory, which allows for energy savings. Sensor scheduling is formulated as a probabilistic belief planning problem. Two algorithms are presented which generate feasible perception schedules: the first is based upon a simple heuristic; the second leverages dynamic programming to obtain optimal plans. Both simulations and real-world experiments on a planetary rover prototype demonstrate over 50% savings in perception-related energy, which translates into a 12% reduction in total energy consumption.**

I. INTRODUCTION

Robots require energy to operate. Yet they only have access to limited energy storage during missions. As we extend the reach of autonomous systems to operate in remote locations, over long distances and for long periods of time, energy considerations are becoming increasingly important. To date, these considerations are often brought to bear in schemes where trajectories or speed profiles are optimised to minimise the energy required for actuation (see, for example, [1], [2], [3]). Here we take a different, yet complementary, approach in considering the energy expenditure for sensing (and, implicitly, computation) associated with navigation. In particular, our goal is to activate the perception system only as required to maintain the vehicle within a given margin around a predetermined path. As the main navigation sensors are switched off and the robot reverts to a lower-powered, less accurate odometry source for parts of the trajectory, any associated computation will also be reduced, leading to further savings in energy.

Naively, such *perception schedules* could be constructed by switching sensors on and off randomly or according to, for example, a fixed frequency. This does, however, suffer the drawback that no heed is paid to drift in the robot's position with respect to the original trajectory: it may not be desirable to deviate by more than an allowed margin from the predetermined route. This arises, for example, in a planetary exploration scenario when conducting long traverses over featureless terrain. Other possible considerations include traversability, obstacles, and the robustness of the localisation system to deviations from the original path. Such naive approaches would also need to be tuned to individual trajectories as savings would depend significantly on trajectory shape. In this work we present two approaches which explicitly account for drift and trajectory shape (though the full trajectory need not be available a-priori) as well as the time required to power-cycle the perception system in order to compute perception schedules (see Figure 1 for a typical schedule computed). Both methods are probabilistically bounded to limit path deviations to a user-specified value. To the best of our knowledge this is the first work to investigate sensor scheduling in the context of robot navigation as a means to save energy during operation. Specifically, beyond exploiting localisation accuracy for energy management, the contributions of this work are

Authors are members of Mobile Robotics Group, University of Oxford, United Kingdom {ondruska, corina, letizia, chi, ingmar}@robots.ox.ac.uk.

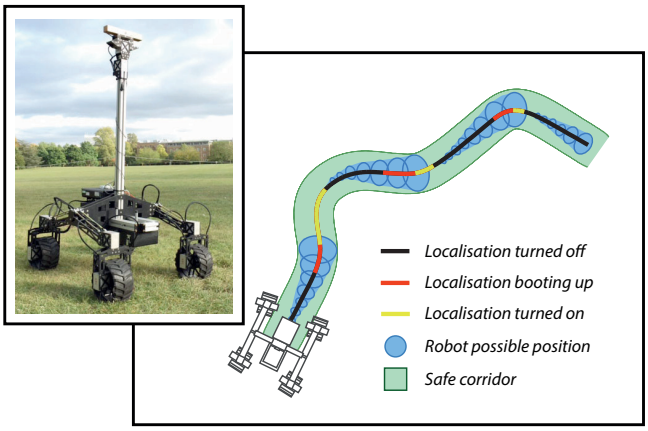

Fig. 1. Example of an energy-optimal perception plan as generated by our method in a path-following scenario. The robot needs to run the sensor only in selected parts of the trajectory (red and yellow) which guarantees it stays within the specified corridor (green) with 90% confidence. This plan saves 50% of the energy consumed by the localisation system, which is 12% of the total system consumption.

- the framing of the problem as a belief Markov decision process over robot poses;
- the description of a greedy algorithm which employs a simple heuristic that ensures feasibility of the perception schedules;
- the description of a belief planning algorithm which leverages dynamic programming to provide *optimal* perception schedules.

Our work is particularly aimed at scenarios where the energy budget for sensing and computing is commensurate with that of actuation - such as is typically the case for planetary rover missions. Our discussion is therefore framed in this context. Our experimental evaluation is carried out on a commercially available planetary rover system. We demonstrate that scheduling the perception subsystem allows for substantial savings in energy expenditure during a mission.

II. RELATED WORKS

Energy consumption in mobile robotics has come to the attention of a number of researchers in recent years. The bulk of this prior art addresses the energetic cost of robot locomotion. A number of works in this vein consider path length, trajectory shape as well as the properties of different type of terrain to predict expected energy use (e.g. [4], [3], [5], [6]). Notably, in a simulation study, the authors of [7] do so in the context of planetary rovers. Mei et al. [1], [8] analyse the impact of different velocities, providing a control algorithm aimed at maximising the distance travelled. Motor dynamics and control input are considered in [9].

To the best of our knowledge, the idea of scheduling a robot's perception system for navigation remains, in practical terms, largely unexplored. The notion that motor energy – next to sensing and computation – may only account for a fraction of the energy requirements of a platform is demonstrated in [10]. This study also notes that the energy consumption of the sensors is a function of the frequency with which they are deployed. Brateman et al. [11] build on these findings and note that energy can be saved by regulating the clock speed of the processor used for perception computation. Based on this insight, in a simulation study, they develop an algorithm which trades off vehicle velocity against processor clock speed.

Our work is closely related in spirit to both [10] and [11]. In contrast to these works, however, we explicitly develop trajectory-dependent activity schedules for the perception subsystem. Our methods are evaluated using real-world experiments. In order to provide optimal schedules we frame this problem as a belief space planning task, which has a successful track record in mobile robotics in dealing with uncertainty in robot state (e.g. [12], [13], [14]).

III. PERCEPTION SCHEDULING AS BELIEF MDP

In this section we define and formalise the problem of perception scheduling in the context of belief Markov decision processes [15]. Specifically we describe the problem as a set of belief states $\mathcal{B}(x_t)$, a set of actions a_t, a transition function $\mathcal{T}(\mathcal{B}(x_t), a_t)$ and a cost function $E(a_t)$.

Consider a situation where the robot has to follow a particular trajectory φ specified as a sequence of poses such that

$$\varphi = \{\tilde{x}_t | t = t_1, t_2, ..., t_N\}. \tag{1}$$

The total energy E required to traverse the trajectory can be split into the energy required to run the navigation system (sensors and respective processing), E_P, and the energy consumed by rest of the system (without loss of generality we attribute this mainly to the motors), E_M,

$$E = E_P + E_M. \tag{2}$$

Our goal is to minimise the total energy E by minimising E_P while maintaining E_M constant. This is achieved by enabling/disabling the localisation system according to a perception schedule, ς, specified by a sequence of actions, a_t such that

$$\varsigma = \{a_t | t = t_1, t_2, ..., t_N\} \tag{3}$$

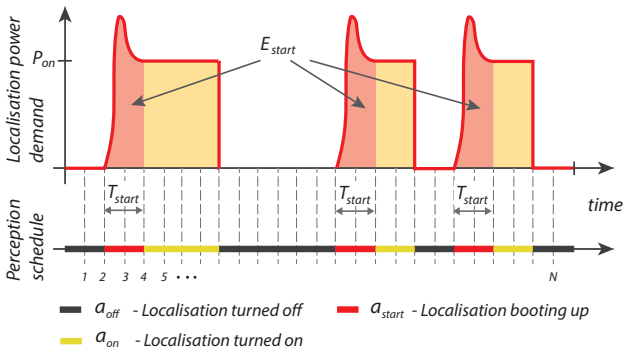

Fig. 2. Energy $E_P(\varsigma)$ required for localisation along the trajectory as given by the perception schedule ς. We assume localisation needs time T_{start} and energy E_{start} to boot and then has a constant power demand P_{on} while it is operational.

where t denotes the time at which an action is taken. These actions correspond to decisions the robot can make with respect to its navigation system. When the localisation system is enabled the robot has a choice to keep it turned on, a_{on}, incurring an energy cost $P_{on} \cdot \Delta t$ where P_{on} is power demand of localisaiton when active, or to turn it off at zero cost. Similarly, when localisation is disabled the robot can keep it turned off, a_{off}, at no cost or boot the localisation system at a cost $E_{start} \cdot \frac{\Delta t}{T_{start}}$ equivalent to an average energy demand over time T_{start} required to turn localisation on. We model E_P simply as a sum of energy demands for individual actions (Figure 2)

$$E_P(\varsigma) = \sum_{t=1}^{N} E(a_t). \tag{4}$$

Once localisation is turned on the robot continuously determines its pose and follows the trajectory. When localisation is off, the robot tries to blindly follow the shape of the trajectory using dead reckoning. In this case, the robot will deviate from the intended trajectory as a result of odometry drift. When localisation is re-enabled the robot observes its position and manoeuvres back towards the path. As discussed in Section V, minor path deviations do not cause an increase in motor energy consumption or hamper the ability to localise. We thus consider the optimisation of E to require only the minimisation of E_P while E_M is independent of the ς. Morever, to limit the magnitude of potential path deviations we consider a set of schedules, $\mathcal{F}$, to be *feasible* only if they result in trajectories contained (with a probabilistic guarantee) within a safe corridor $\mathcal{A}_t$.

Therefore, the task of perception scheduling is to find an optimal, feasible plan ς^* which minimises the energy consumption attributed to the localisation system:

$$\varsigma^* = \arg\min_{\varsigma \in \mathcal{F}} E_P(\varsigma). \tag{5}$$

A. Schedule feasibility

We consider a set of feasible schedules $\mathcal{F}$, such that

$$\mathcal{F} = \mathcal{F}_1 \cap \mathcal{F}_2, \tag{6}$$

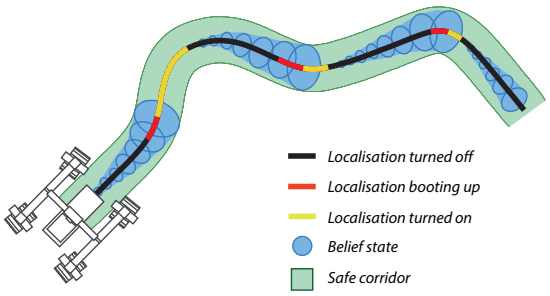

Fig. 3. Belief evolution over the robot's pose x_t around the trajectory φ based on the perception schedule, ς. When localisation is turned on, the position is known and the belief $\mathcal{B}(x_t)$ is given by a Dirac delta function, $\delta(x_t - \tilde{x}_t)$. When localisation is turned off, the belief evolves based on the robot's motion model. The perception schedule is conceived such that the robot always remains in the designated safe corridor area, $\mathcal{A}_t$.

where $\mathcal{F}_1$ and $\mathcal{F}_2$ separately satisfy different feasibility constraints. First we consider that localisation can not be turned on instantaneously but requires time T_{start} to initialise (turning on the sensor, launching an executable, etc). This means a sufficient number of a_{start} actions need to precede the first action a_{on}:

$$\mathcal{F}_1 = \left\{ \varsigma \;\middle|\; \begin{array}{c} a_{t+1} = a_{on} \wedge a_t \neq a_{on} \rightarrow a_{t-t'} = a_{start} \\ \forall t, t' : 0 \leq t' \leq T_{start} \end{array} \right\} \quad (7)$$

Secondly, we want the robot to remain within a safe corridor of the planned trajectory even when localisation is turned off. In this case the real pose x_t is not known. Instead we can compute a belief state $\mathcal{B}(x_t)$, which is a distribution over the possible robot poses in relation to our target, $\tilde{x}_t$:

$$\mathcal{B}(x_t) = p(x_t | \tilde{x}_{1:t}, a_{1:t}). \quad (8)$$

This belief evolves based on the actions taken and can be encoded in the form of a transition function $\mathcal{T}$. When the localisation is turned on we approximate this belief by a Dirac delta function centred on the intended pose. This reflects an assumption that localisation is exact. As the camera is turned off, the belief over poses evolves based on the command input u_t used to drive the robot from pose $\tilde{x}_t$ to $\tilde{x}_{t+1}$ along the trajectory as well as on the odometry noise model (Figure 3.)

$$\mathcal{T}(\mathcal{B}(x_t), a_t) = \begin{cases} \delta(x_t - \tilde{x}_t) & \text{if } a_t = a_{on} \\ p(x_t | \mathcal{B}(x_{t-1}), u_t) & \text{otherwise.} \end{cases} \quad (9)$$

As illustrated in Figure 4, every perception schedule $\{a_1, a_2, ..., a_N\}$ corresponds to a sequence of belief states $\{\mathcal{B}(x_1), \mathcal{B}(x_2), ..., \mathcal{B}(x_N)\}$. We define the safe obstacle-free area around the robot's intended pose $\tilde{x}_t$ at time t as $\mathcal{A}_t$. The desired behaviour is that the robot will not exit this zone. Due to the randomness of the motion process when localisation is turned off, the risk of the robot leaving this corridor can not be completely avoided. Instead, we leverage a probabilistic guarantee that the true robot pose is contained within this area with probability greater than or equal to p_{conf}. We are therefore interested in the set of solutions satisfying, at all times

$$\mathcal{F}_2 = \left\{ \varsigma \;\middle|\; \int_{\mathcal{A}_t} \mathcal{B}(x_t) dx_t \geq p_{conf} \quad \forall t \right\}. \quad (10)$$

IV. SOLUTION METHOD

In this section, we describe the two algorithms used for computing feasible perception schedules. The first one, the *greedy* algorithm, is simple to implement: it does not require knowledge of the full trajectory in advance and, in practice, can save a substantial amount of perception-related energy. However, this algorithm may produce suboptimal behaviour in some situations as a result of taking decisions which are only locally optimal. The second algorithm overcomes this limitation by considering the entire trajectory (or up to a given planning horizon) and presents an efficient belief planning implementation based on dynamic programming [16]. This algorithm is able to find the *optimal* trajectory through the belief space attainable by the robot which, in turn, correspond to the optimal perception schedule, ς^* (see Fig. 4).

A. Greedy Algorithm

A simple procedure to obtain a valid plan is to keep the sensor off as long as possible while simultaneously modelling the robot's belief state $\mathcal{B}(x_t)$. When the uncertainty on the state becomes too high, i.e. $\int_{\mathcal{A}_t} \mathcal{B}(x_t) dx_t < p_{conf}$, we turn the sensor on, localise and turn the sensor off again. The only constraint is related to the boot time of the sensor, T_{start}. This means that we need to simulate the evolution of the belief state in advance for a time horizon T_{start} in order to be able to start the sensor in time. It also means that we always need to know the trajectory shape only for

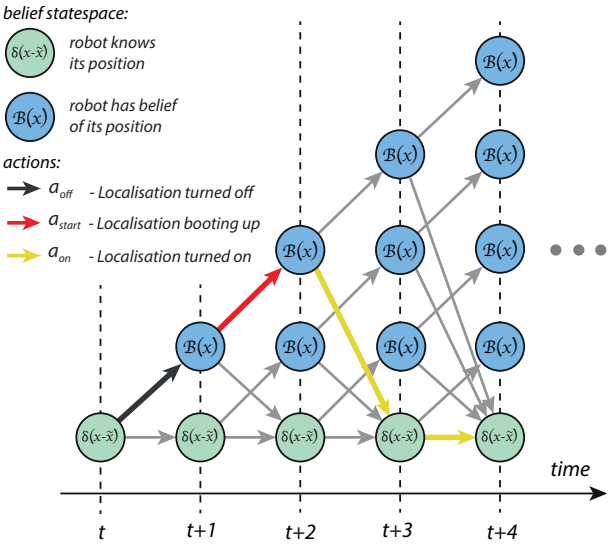

Fig. 4. The belief space attainable by the robot starting at time t and a possible perception schedule indicated by the actions performed (coloured arrows). At every time step the robot has a choice to either localise or not. The latter results in more uncertainty over its current pose. Localisation has an energy cost associated with it but remains the only mechanism by which pose uncertainty can be limited.

Algorithm 1 Greedy Control Strategy

Input:	$\mathcal{B}(x_1)$	*current belief state - see Equ. 9.*
	T_{start}	*time required to localise*
	$\tilde{x}_{1:T_{start}}$	*trajectory for T_{start} seconds*
	$\mathcal{A}_{1:T_{start}}$	*safe corridor for T_{start} seconds*
	p_{conf}	*confidence of staying in the corridor*
Output:	a_1	*perception action taken*

Forward simulate belief state evolution.

1. **for** $t = 2, 3, ..., T_{start}$
2. $\quad \mathcal{B}(x_t) = p(x_t, \mathcal{B}(x_{t-1}), u_t)$
3. $\quad$ **if** $\int_{\mathcal{A}_t} \mathcal{B}(x_t) dx_t < p_{conf}$ **then**
4. $\quad\quad$ **return** $a_1 \leftarrow \begin{cases} a_{on} & \text{if } \mathcal{B}(x_1) = \delta(x_1 - \tilde{x}_1) \\ a_{start} & \text{otherwise} \end{cases}$
5. **return** $a_1 \leftarrow a_{off}$

the next T_{start} seconds. The principle can be efficiently implemented as an online perception controller as shown in Algorithm 1. In such cases, the perception schedule ς is not created explicitly but is generated on-the-fly as a result of the controller's decisions. This simple procedure is surprisingly effective. As described in section V, it can save over 40% of the perception-related energy consumption on the robotic platform used in this work.

However, if more complex trajectories are followed, this behaviour can be suboptimal. Consider the scenario shown in Fig. 5 a). When performing a motion resulting in high pose uncertainty (e.g. turning), keeping the sensor on until the motion is finished results in higher energy savings since it prevents the later need for relocalisation. Similarly, in Fig. 5 b), if the energy to localise (E_{start}) is low but the interval T_{start} is long, it would be optimal to localise early, at t_1, and then turn the sensor off. Waiting until t_2 is suboptimal and requires more energy, since the interval $t_3 - t_2$ is shorter than the time required to boot the system (T_{start}) and the sensor would need to be kept on.

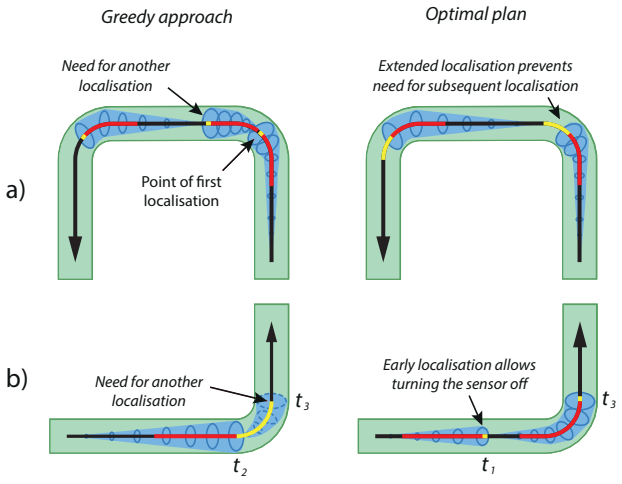

Fig. 5. Example scenarios when a locally optimal decision results in a globally suboptimal solution. A detailed description is provided in the text.

B. A Belief Planning Solution

Given knowledge of the robot's initial position as well as of the entire trajectory, the optimal perception schedule can be computed as the energetically cheapest path to any belief state, $\mathcal{B}(x_N)$, through all attainable belief states, $\mathcal{B}(x_1), \mathcal{B}(x_2), ..., \mathcal{B}(x_N)$. It is important to observe that even though there are infinitely many possible belief states, our robot can visit at most $\frac{N(N+1)}{2}$ different ones. These states are produced by turning off the sensor at time t, and continuing blind for the remainder of the trajectsory. This bound is important because it indicates that all possible belief states can be precomputed which in turn allows the deployment of deterministic planning algorithms commonly used for *finite* Markov decision processes [17]. This results in the optimal schedule $\varsigma^*(\mathcal{B}(x_t))$ yielding minimal energy usage, $E^*(\mathcal{B}(x_t))$ for any starting state, $\mathcal{B}(x_t)$. The optimal schedule for the whole trajectory, ς^*, is equal to $\varsigma^*(\delta(x_1 - \tilde{x}_1))$.

However, computing all possible belief states in advance is inefficient. Instead we employ a dynamic programming approach to compute optimal plans at each timestep, $\varsigma_t^* = \varsigma^*(\delta(x_t - \tilde{x}_t))$, as well as the associated energy consumption, $E_t^* = E^*(\delta(x_t - \tilde{x}_t))$, only for belief states $\delta(x_t - \tilde{x}_t)$ in order of $t = N, N-1, ..., 1$. This results in an efficient solution as detailed in Algorithm 2. Reachable belief states are computed only when required and previously determined optimal values are reused. The algorithm has time complexity $\mathcal{O}(N^2 \cdot T(\mathcal{B}))$ and memory complexity $\mathcal{O}(N + M(\mathcal{B}))$, where $T(\mathcal{B})$ is the time and $M(\mathcal{B})$ is memory complexity corresponding to the belief state evolution (check lines 2-3 of Algorithm 1 and 7-8 of Algorithm 2). Our implementation uses a particle filter to represent the belief states, which results in time complexity $\mathcal{O}(N^2 L)$ and memory complexity $\mathcal{O}(N + L)$, where L is the number of particles.

V. EXPERIMENTS

We evaluate the performance of the algorithms presented using a commercially available planetary rover prototype in a path-following task with a total driving distance of $1.8km$. We demonstrate that the methods described are able to save over 50% of the sensing-related energy, which constitutes approximately 12% of the overall energy usage of the rover.

A. System

The robotic platform used in these experiments is the ARC Q14 planetary rover shown in Figure 6. It is designed to mimic the configuration and specifications found on rovers deployed for planetary exploration. In particular, robot locomotion is provided by eight low-power motors, independently articulating steering and four rotating wheels, traveling at a maximum speed of $0.5m/s$. The robot is equipped with a Point Grey Bumblebee XB3 camera for visual localisation and wheel odometry sensors aiding motor controllers. The computation is carried out on an on-board MicroSVR computer containing an Intel Core i7 processor and 16GB RAM. The robot is equipped with current sensors, which allow the measurement of the power consumption of individual motors. This was found to be of the order of

Algorithm 2 Optimal plan computation.

Input: $\tilde{x}_{t_1:t_K}$ *trajectory*
$\mathcal{A}_{t_1:t_K}$ *safe corridor*
P_{on} *power demand of perception*
T_{start} *time required to localise*
E_{start} *energy required to localise*
p_{conf} *confidence of not leaving the corridor*
Output: ς^* *perception plan*

Backward pass - compute E_t^* and ς_t^* for each t
for $t = N-1, N-2, ..., 1$
  Value of keeping the sensor on
  $E_t^* \leftarrow \Delta t \cdot P_{on} + E_{t+1}^*$
  $\varsigma_t^* \leftarrow \{a_{on}, \varsigma_{t+1}^*\}$
  Turn the sensor off and turn it on at time t'
  $\varsigma \leftarrow \{\}$
  $\mathcal{B}(x_t) \leftarrow \delta(x_t - \tilde{x}_t)$
  for $t' = t+1, t+2, ..., N$ **do**
    Simulate belief state
    $\mathcal{B}(x_{t'}) \leftarrow p(x_{t'}, \mathcal{B}(x_{t'-1}), u_{t'})$
    if $\int_{\mathcal{A}_{t'}} \mathcal{B}(x_{t'}) dx_i < p_{conf}$ **then**
      break
    if $t' < t + T_{start}$ **then**
      $\varsigma \leftarrow \{a_{start}, \varsigma\}$
    else
      Vale of turning the sensor on at time t'
      if $E_{start} + E_{t'}^* < E_t^*$ **then**
        $E_t^* \leftarrow E_{start} + E_{t'}^*$
        $\varsigma_t^* \leftarrow \{\varsigma, a_{on}, \varsigma_{t'}^*\}$
        $\varsigma \leftarrow \{a_{off}, \varsigma\}$
return $\varsigma^* = \varsigma_1^*$

$30W$. Nominal values specified by the manufacturers are used for the CPU ($25W$) and the camera ($5W$). This amounts to an average overall power consumption of $60W$ while moving and localising. We estimate energy savings when the localisation system is turned off to be of the order of $10W$. This comprises $5W$ (the nominal value) for the camera as well as an additional $5W$ estimated reduction in CPU power.

The navigation system is based on visual teach and repeat (T&R) [18], [19], which is often employed in the space domain and similar approach can used for example in sample and return missions [20], [21]. T&R enables a mobile robot to autonomously follow a previously driven path with high accuracy. First, a trajectory φ is driven by a human operator. During this teach run the system builds a feature map used later for localisation. Next, the perception schedule ς is computed using the presented algorithms. Finally, the robot attempts to follow the path autonomously, localising in the map according to the perception schedule. The entire system architecture is illustrated in Figure 7. In order to evaluate the robot's actual trajectory when adhering to perception schedules the camera was not switched off in reality. Sensor deactivation was instead simulated by cutting off the image stream in software and subtracting the appropriate energy

Fig. 6. The robotic platform used for experiments. The robot is a commercially-available version of a planetary rover. It is equipped with low-power motors, a stereo camera pair for visual localisation and a CPU to carry out the computation.

values from the total for the entire system to obtain the reduced energy demand. Note that, although we tested the behaviour and the performance of our methods in a T&R context, this is not the only scenario in which it is possible to use these methods. The utility of perception schedules during path following is general and does not depend on the source of the trajectory.

B. Perception schedule computation

To generate the perception schedules we considered localisation to be reliable if the distance from the robot's real pose, x_t, to the expected one, $\tilde{x}_t$, is less than $\gamma = 0.9m$ and if the angular error is less than $20°$. This defines the scope of the safe area, $\mathcal{A}_t$. Similarly, we assumed that the robot can reliably localise after time $T_{start} = 4s$ and that the energy consumption required to start up the localisation system is $E_{start} = T_{start} \cdot P_{on} = 0.0111Wh$.

We used a standard odometry motion model [15] to approximate the robot's drift when the localisation system is not active. In this model, let $(\Delta x, \Delta y, \Delta\theta)_t$ be the command input fed to the controller required to move the robot from state $\tilde{x}_t$ to $\tilde{x}_{t+1}$. The robot's motion during this interval is

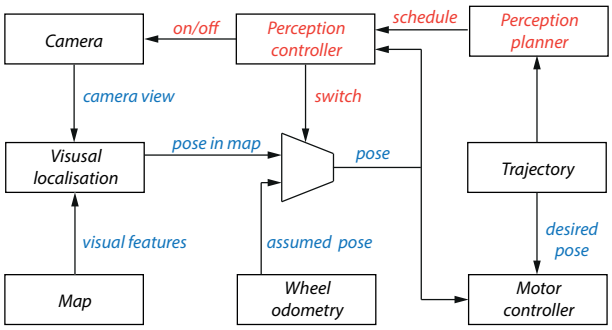

Fig. 7. System diagram enabling perception scheduling. The localisation system feeds pose information into a standard trajectory-control pipeline. The perception controller turns the sensor on or off based on the pre-computed perception schedule.

approximated as a composition of an initial rotation, ϕ_1, an intermediate translation, τ, and a second rotation ϕ_2, as shown in Figure 8. These motions are described by

$$\phi_1 = atan2(\Delta y, \Delta x), \tag{11a}$$
$$\tau = \sqrt{\Delta x^2 + \Delta y^2}, \tag{11b}$$
$$\phi_2 = \Delta\theta - \phi_1. \tag{11c}$$

We assume that the real motion is affected by additive zero-mean Gaussian noise with standard deviation proportional to the motion magnitude:

$$\tilde{\phi}_1 \sim \mathcal{N}(\phi_1, \alpha_1|\phi_1| + \alpha_2\tau), \tag{12a}$$
$$\tilde{\tau} \sim \mathcal{N}(\tau, \alpha_3\tau + \alpha_4(|\phi_1| + |\phi_2|)), \tag{12b}$$
$$\tilde{\phi}_2 \sim \mathcal{N}(\phi_2, \alpha_1|\phi_2| + \alpha_2\tau). \tag{12c}$$

where the tuning parameters $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ govern the influence of each of the motion components. Using these definitions, the real robot motion is given by

$$\Delta\tilde{x} = \tilde{\tau}\ cos(\tilde{\phi}_1), \tag{13a}$$
$$\Delta\tilde{y} = \tilde{\tau}\ sin(\tilde{\phi}_1), \tag{13b}$$
$$\Delta\tilde{\theta} = \tilde{\phi}_1 + \tilde{\phi}_2. \tag{13c}$$

In our experiments, we obtained the parameter vector α, by maximising the log likelihood of the robot motion, as reported by the visual localisation system, given the commands used to control the robot on a calibration trajectory of length $85m$. The resulting values are given in Table I.

In the implementation of the perception scheduling algorithms, we made use of the particle filter described in [15], with $L = 10000$ particles, to represent the belief state. The propagation of the belief and the computation of the probability of the robot being in the safe area $\mathcal{A}_t$ (lines 2,3 of Algorithm 1 and 7, 8 of Algorithm 2) is then obtained by sampling the described motion model and computing the number of particles in $\mathcal{A}_t$. For a typical trajectory of length $N = 500$ timesteps, the generation of the perception schedule is typically obtained within $0.1s$ in the case of the Greedy algorithm and within $5s$ in the case of the Belief planning algorithm.

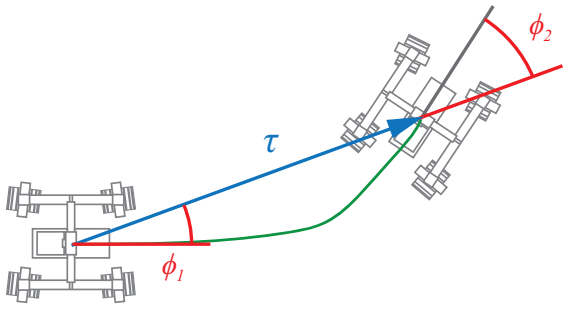

Fig. 8. The plant model used to describe rover motion [15]. The robot motion (green) in a time interval is approximated by a rotation, ϕ_1, followed by a translation, τ, and a second rotation, ϕ_2. The rotations and the translation are affected by noise.

α	*Odometry noise model*	$[0.428, 0.100, 0.054, 0.150]$
P_{on}	*Localisation power demand*	$10W$
T_{start}	*Localisation boot time*	$4\ s$
E_{start}	*Localisation boot energy*	$0.0111\ Wh$
p_{conf}	*Confidence threshold*	$90\ \%$
γ	*Corridor width*	$0.9\ m$

TABLE I

PARAMETERS USED TO GENERATE THE PERCEPTION SCHEDULES IN THE EXPERIMENTS.

C. Evaluation

We evaluate the performance of the algorithms on three different trajectories of length $61m$, $54m$ and $53m$, as depicted in Figure 9. These trajectories were chosen to consider different terrain conditions. The first trajectory was driven on asphalt in an urban area, the second on gravel paths and the third on grass.

To establish a baseline for the energy consumption, the robot followed all trajectories K times with the localisation system active for the entire traverse. Then, for each trajectory, perception schedules were generated using both the *greedy* and the *belief planning* algorithm. Each perception schedule was executed autonomously by the robot K times. The schedules are shown in Figure 9. The first two trajectories were travelled $K = 3$ times each, and the last trajectory $K = 5$ times.

All trajectories driven when executing the perception schedules remained within the safe area as required by the feasibility constraint. For each trial, the perception related energy, E_P, the motor power consumption, E_M, the proportion of the saved perception related energy, ΔE_P, and the total saved energy ΔE were recorded and compared to the power consumption when localising throughout the traverse. The mean values over the K trials for each trajectory are summarised in Table II. In Figure 10, we provide details on part of the third trajectory, illustrating the actual path traversed by the robot following the schedule produced by the *Belief Planning* algorithm. On average we can see

	Perception schedule	K	E_M [Wh]	E_P [Wh]	ΔE_P [%]	ΔE [%]
Trajectory I	All-time	3	3.1091	0.5140	0	0
	Greedy	3	3.0247	0.2452	54.15	9.75
	Belief planning	3	2.9894	0.2165	58.44	11.52
Trajectory II	All-time	3	2.3364	0.4203	0	0
	Greedy	3	2.2284	0.2131	48.88	9.42
	Belief planning	3	2.2136	0.1903	53.59	12.80
Trajectory III	All-time	5	2.9494	0.4565	0	0
	Greedy	5	2.7886	0.2794	38.73	9.92
	Belief planning	5	2.8136	0.2459	46.38	10.10

TABLE II

ENERGY USAGE ALONG THE THREE TEST TRAJECTORIES. EACH TABLE CORRESPONDS TO ONE TRAJECTORY AND SHOWS THE ENERGY USED BY THE MOTORS E_M, THE CAMERA E_P, THE PROPORTION OF SAVED PERCEPTION RELATED ENERGY ΔE_P AND TOTAL ENERGY ΔE.

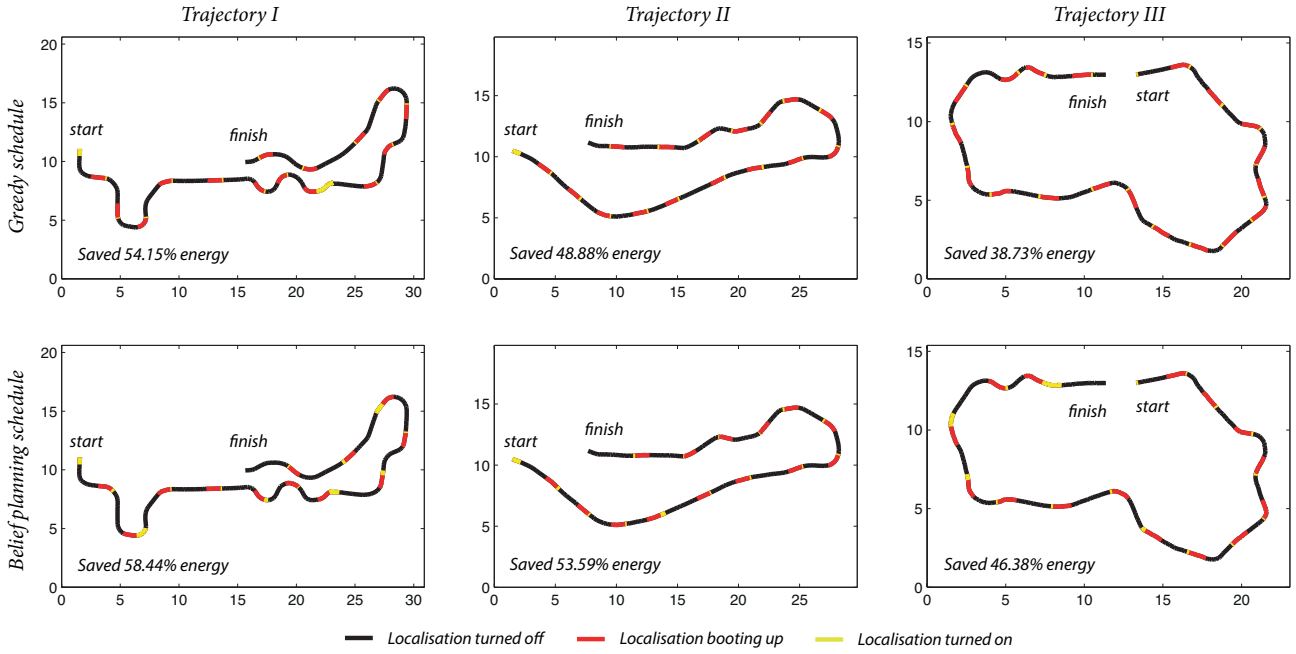

Fig. 9. Perception schedules for three different trajectories and respective perception energy savings ΔE_P, generated by the *Greedy algorithm* (top row) and the *Belief planning algorithm* (bottom row).

that the *greedy* algorithm was able to save around 47.3% of the energy associated with sensing (9.7% of the total energy usage) while employing the *Belief Planning* algorithm resulted in a reduction in perception energy of 52.8% (11.5% of the total).

If we examine Figure 9 in further detail, we can observe that the shape of the trajectory plays an important role in the energy savings. In particular, the straight segments of the trajectories require localisation less often than the ones characterised by turns. We also notice that the *Belief Planning* method exhibits the optimised behaviour described in Figure 5. Our assumption that the execution of the perception schedule did not cause an increase in motor energy consumption has been validated empirically during these trials. Surprisingly, we observed the opposite in our experimental results. The motor energy was slightly lower when following the perception schedules. We explain this by the fact that the robot performed fewer feedback corrections when the localisation system was turned off.

It is also interesting to observe that even though the second and the third trajectory have similar length, the energy consumed by the motors is higher for the second trajectory. We attribute this to the difference in terrain (grass vs. gravel paths), and the more complex trajectory. This requires the camera to be switched on for longer for both algorithms.

Finally, we performed a simulation-based analysis of the behaviour of the planning algorithms when varying key system parameters. Figure 11. shows the proportion of the perception-related energy saved, ΔE_P, after changing the allowed corridor width γ, the time necessary for the localisation system to boot, T_{start}, and the confidence threshold, p_{conf} for the first trajectory. Increasing the confidence threshold as well as tightening the corridor or decreasing the boot time produces a noticeable decrease in energy savings

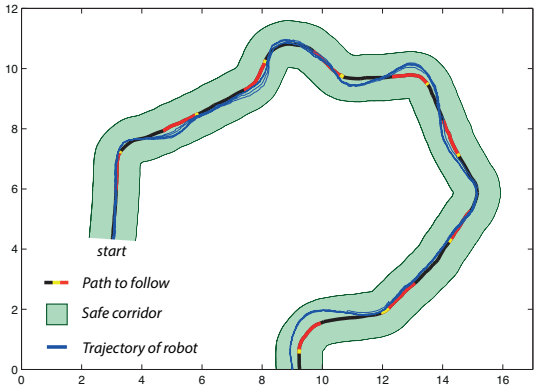

Fig. 10. Part of the second teach run and the actual trajectories taken by the robot while following the perception schedule. While the robot does not follow the trajectory exactly, it never leaves the safe area [green].

for both algorithms. When increasing the camera boot time the *greedy* algorithm performs progressively worse than the optimal *Belief Planning* algorithm. This is attributed to the fact that longer boot times have a bigger impact on the schedule for even slightly complex trajectories.

VI. CONCLUSION

This paper explores the efficacy of scheduling robot perception in order to save energy when following a planned path. This is fundamentally different to the more common approach of minimising motor energy by altering the robot's trajectory. Our perception scheduling problem is framed as a belief space planning task which explicitly accounts for localisation startup cost as well as pose uncertainty while the main localisation system is turned off. This allows for both optimal schedules to be computed and performance guarantees to be provided regarding the allowable deviation

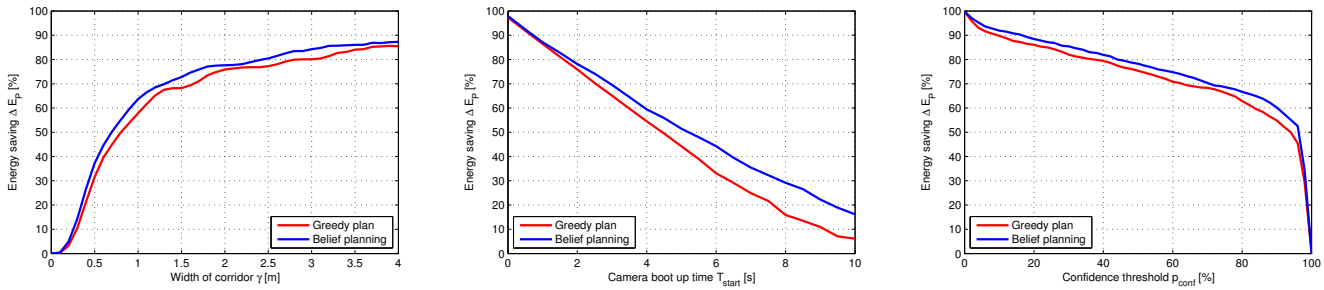

Fig. 11. Perception energy savings ΔE_P as a function of the corridor width γ, the camera booting up time T_{start} and the confidence threshold p_{conf}. Increasing the confidence threshold as well as tightening the corridor or decreasing the camera booting time, produces a noticeable decrease of the energy savings for both algorithms.

from the intended trajectory. Using $1.8km$ of autonomous traversals our perception schedules are demonstrated to save a maximum of 58% of the perception energy, which accounts for 11.5% of the total robot energy expenditure. This is in comparison to a simpler greedy approach, which saves 5.5% less energy on perception on average than the optimal solution based on dynamic programming. This is a function of trajectory shape as well as the robotic platform used. A more in-depth characterisation of this dependency, together with intelligent detection and handling of situations if the robot exits the safe corridor, is subject to further work. In all scenarios greater accuracy can be obtained with more precise models of robot drift and energy usage. Other avenues for further investigation include an intelligent the joint optimisation of robot trajectory and perception schedule in order to achieve even higher energy savings. This will need to be balanced with our surprising experimental observation that the reduced feedback from when the perception system is disabled can result in lower motor energy consumption.

ACKNOWLEDGMENTS

The authors would like to gratefully acknowledge support of this work by the UK Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/K034472/1, European Community's Seventh Framework Programme under grant agreement no FP7-610603 (EUROPA2) and via the DTA scheme as well as by the UK Space Agency (grant number ST/L002981/1) as part of the CREST-2 Initiative. We would also like to thank Winston Churchill and Geoff Hester for software support.

REFERENCES

[1] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "Energy-efficient motion planning for mobile robots," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 5. IEEE, 2004, pp. 4344–4349.

[2] J. J. Biesiadecki, P. C. Leger, and M. W. Maimone, "Tradeoffs between directed and autonomous driving on the mars exploration rovers," *The International Journal of Robotics Research*, vol. 26, no. 1, pp. 91–104, 2007.

[3] G. Ishigami, K. Nagatani, and K. Yoshida, "Path planning and evaluation for planetary rovers based on dynamic mobility index," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 601–606.

[4] Z. Sun and J. Reif, "On energy-minimizing paths on terrains for a mobile robot," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 3. IEEE, 2003, pp. 3782–3788.

[5] J. A. Broderick, D. M. Tilbury, and E. M. Atkins, "Characterizing energy usage of a commercially available ground robot: Method and results," *Journal of Field Robotics*, vol. 31, no. 3, pp. 441–454, 2014.

[6] S. Martin and P. Corke, "Long-term exploration and tours for energy constrained robots with online proprioceptive traversability estimation," in *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 5778–5785.

[7] S. Fallah, B. Yue, O. Vahid-Araghi, and A. Khajepour, "Energy management of planetary rovers using a fast feature-based path planning and hardware-in-the-loop experiments," *Vehicular Technology, IEEE Transactions on*, vol. 62, no. 6, pp. 2389–2401, 2013.

[8] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "Deployment of mobile robots with energy and timing constraints," *Robotics, IEEE Transactions on*, vol. 22, no. 3, pp. 507–522, 2006.

[9] C. H. Kim and B. K. Kim, "Energy-saving 3-step velocity control algorithm for battery-powered wheeled mobile robots," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 2375–2380.

[10] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "A case study of mobile robot's energy consumption and conservation techniques," in *Advanced Robotics, 2005. ICAR'05. Proceedings., 12th International Conference on*. IEEE, 2005, pp. 492–497.

[11] J. Brateman, C. Xian, and Y.-H. Lu, "Energy-effcient scheduling for autonomous mobile robots," in *Very Large Scale Integration, 2006 IFIP International Conference on*. IEEE, 2006, pp. 361–366.

[12] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 4, pp. 566–580, 1996.

[13] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *The International Journal of Robotics Research*, 2009.

[14] J. Van Den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1263–1278, 2012.

[15] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[16] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 2012.

[17] A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 1998.

[18] P. T. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of Field Robotics, Special issue on "Visual mapping and navigation outdoors"*, vol. 27, no. 5, pp. 534–560, May 2010.

[19] C. McManus, P. T. Furgale, B. Stenning, and T. D. Barfoot, "Lighting-invariant visual teach and repeat using appearance-based lidar," *Journal of Field Robotics*, vol. 30, no. 2, pp. 254–287, 2013.

[20] N. R. C. Committee on the Planetary Science Decadal Survey, *Vision and Voyages for Planetary Science in the Decade 2013-2022*. Washington, D.C.: National Academies Press, 2011.

[21] R. E. Arvidson, "Planetary science decadal survey: Mars 2018 MAX-C caching rover," National Aeronautics and Space Administration (NASA), Tech. Rep., March 2010.