Sustainable Robot Foraging: adaptive fine-grained multi-robot task allocation for Maximum Sustainable Yield of biological resources

Zhao Song and Richard T. Vaughan*

***Abstract*— We introduce the concept of Maximum Sustainable Yield (MSY) to the context of autonomous robot foraging. MSY is an optimal approach to the problem of maximizing sustainable foraging where the resources harvested are replenished by logistic growth, e.g. living things. Over-harvesting reduces both the instantaneous resource availability and growth rate, and above some threshold will permanently deplete resources. Under-harvesting is sustainable, but fails to maximally exploit the resources. We describe a system model and use it to determine the optimal allocation of robot work to resource-producing 'patches'. We give a practical illustration of a troublesome feature of MSY: it is too sensitive for a fixed allocation to be sustainable in practice. We show how to centrally allocate a number of robots to each patch, and then locally adapt the work rate of each robot to achieve sustainable and near-optimal foraging. This is the first study of robot foraging where the robots' activity modifies the productivity and sustainability of the environment.**

I. Introduction: SuRF

We introduce the Sustainable Robot Foraging (SuRF) problem, in which one or more robots must maximize the long-term profit obtained by harvesting self-reproducing resources from the environment. For foraging to be indefinitely *sustainable* the resource-generating population must never be destroyed by over-harvesting, while under-harvesting fails to maximally exploit resources. This is a fundamental problem for living or artificial systems that aim to exploit biomass resources for long periods.

Population growth over time is modeled using the classical logistic function proposed by Verhulst [1] to model animal populations, and since applied to the growth of tumors and many other natural systems. The logistic model improved on the earlier exponential growth model of Malthus [2] by recognizing that populations generally can not grow unbounded, with growth limited as resources consumed by the existing population become scarce. A formula for obtaining the optimal harvest rate in systems with logistic growth was first obtained in an effort to maximize fish catches [3], and became well known in this context as the Maximum Sustainable Yield.

To apply these insights to the robotics context, we investigate the classical foraging problem in which autonomous mobile robots must collect *pucks*; generic atomic objects of value to the robots' owner. Pucks are not distributed at random in the environment, but exist in areas of locally high density called *patches* per the behavioural ecology literature [4]. The number of pucks in a patch (the *patch size*) changes over time according to the logistic function, simulating a naturally regrowing resource that is harvestable in discrete units, such as mushrooms, acorns, fruits, animals and fish.

*School of Computing Science, Simon Fraser University, British Columbia, Canada. {zhaos,vaughan}@sfu.ca

Fig. 1: Robots forage for resources that demonstrate logistic population growth. To obtain maximum sustainable profit, the robots must harvest resources at the rate that maximizes the rate of regrowth. This is the Maximum Sustainable Yield. [Artwork © Christine Larson]

Once collected, pucks must be delivered to a central collecting point, at which time the robot system is credited with one unit of reward. Our goal is to maximize the total reward obtained by the system. If the reward per unit of resource is constant or discounted only slightly over time, then the optimal policy is to permanently sustain foraging while maximizing the instantaneous reward rate [5], [6]. To achieve this, robots must harvest resources from each patch at the rate that provides the fastest resource growth rate at that patch. This implies that the patch will remain at some ideal population size. Collect pucks too slowly and the patch is sustainable but less than optimally productive. Collect pucks too quickly and the patch population inevitably dwindles. Once below a critical population size (e.g. two mammals) a patch can not self-generate and is permanently unproductive.

The paper proceeds as follows. First we describe the Maximum Sustainable Yield formulation and use to find the optimal robot work allocations for our robot foraging problem. Realizing the model in a numerical simulation, we observe a well-known problem with MSY: the system is dynamically sensitive to small perturbations, so that the fixed allocation can not provide good sustainable foraging in practice. To cope with this we describe and demonstrate a simple feedback controller that locally modulates the foraging rate at each patch to achieve sustainability and close to optimal performance. We demonstrate the controller achieving sustainable and near-optimal foraging in a simple robot simulator.

The main contributions of this paper are (i) the introduction of robot foraging where the robots' activity modifies the future productivity and of the environment; (ii) introducing the MSY concept into a robot setting to define the SuRF problem; (iii) a multi-resolution task allocation method whereby an integer number of robots is over-allocated to a task and each robot adaptively reduces its individual work rate to achieve an effective work allocation with sub-robot resolution.

II. Related Work

An excellent but dated overview of collective robotics is given by [7]. Lerman *et al* [8] describe a mathematical model of the dynamics of collective behaviors in foraging system and applied it to study adaptive task allocation in mobile robots. Puck foraging has become a canonical task in multi-robot systems; for example [9] examines the energy consumption of multiple foraging robots when pucks are dynamically generated at random.

The relation between population growth rate and population density has been discussed since the nineteenth century. Alternative mathematical models have been proposed, e.g. [10], [11], and sigmoidal growth curves have been shown to better match some observations of bacterial growth [12]. Despite these challenges, the logistic model is still widely used in population studies.

The logistic model has been utilized in multi-robot systems, for example applied to the optimization of communication [13]. Perhaps the most similar previous work is by Tereshko *et al*, who considered a bee colony as dynamical system gathering information from an environment and adjusting its behavior in accordance to it [14]. They also used the logistic model to analyse the population of foragers. They did not consider the sustainability of foraging; the foragers chose the most productive patch without attempting to maintain its productivity, in contrast to our paper.

III. Logistic Population Growth

Biological populations tend to grow exponentially when not constrained by resource availability, but growth rates decrease as resources become scarce, until some maximum population size is reached. Verhulst originally proposed the differential equation

$$\frac{dP}{dt} = rP(1 - \frac{P}{K}) \tag{1}$$

to model a system with self-limiting growth, where K is the maximum supported population: the "carrying capacity" of the environment, and r is the unconstrained population growth rate [1]. It follows that the population at time t can be found when r, K and the minimum population

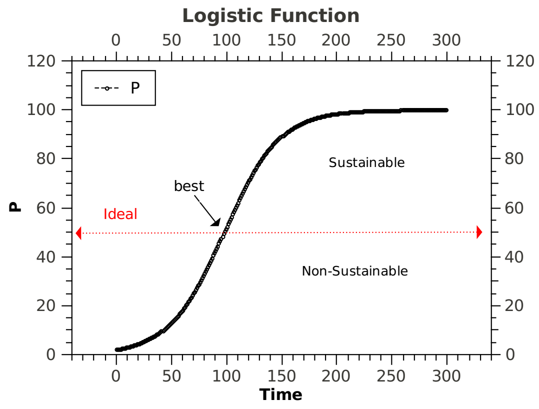


Fig. 2: Evolution of the logistic function, Eq. 1. Showing population size P over time without harvesting, starting with $P_0 = 1$ until carrying capacity K is approached at around $t = 300$. The point of steepest gradient is indicated "best". At this population size, the population increases at the fastest rate. The Maximum Sustainable Yield is achieved by harvesting such that the population remains at this ideal size over time, here indicated by the "ideal" line $P = 50$.

bound P_0 are known[1]. Unless the population is reduced by some external event, the population eventually grows to the carrying capacity.

$$P(t) = \frac{K}{1 + \frac{K - P_0}{P_0} e^{-rt}} \tag{2}$$

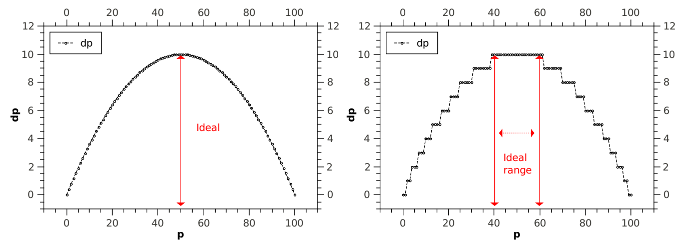
$$\lim_{t \to \infty} P(t) = K \tag{3}$$

An example evolution of the system from $P_0 = 1$ with $K = 100, r = 0.04$ is shown in Figure 2. Note that the population size increases to asymptotically approach K, but that the growth rate changes constantly, with a peak at around $t = 100$ at a population $P = 50$. Harvesting such that the population is maintained at 50 will maximize the sustainable yield.

A. Patch destruction

If a population size becomes smaller than P_0 e.g. due to harvesting, the population can never increase and will decline to zero. This is a serious practical problem, since if we are to guarantee to avoid this with a fixed work rate we need to know all the logistic model parameters with perfect accuracy. In practice, we can supply robots with a conservative minimum population threshold below which they will not collect a puck. If the threshold is too high we may under-harvest, but sustainably. If it is too low we inevitably destroy the patch in finite time.

[1] P_0 is conventionally written $P(0)$: we rename it here avoid confusion with the population size $P(t)$ at time $t = 0$, since in our experiments populations need not be initialised to P_0.



(a) Continuous harvesting (b) Discrete harvesting

Fig. 3: A plot of the population growth rate against population size. ($dp = \Delta p(t)$). In the continuous version (left) there is a unique population size that maximizes the growth rate, where the slope of the curve is zero. In the discrete version (right), there is a range of populations that produce the optimal growth rate.

B. Discretization

For simplicity of implementation we use the discrete-time version of Equation 1:

$$P(t+1) = P(t) + rP(t) - \frac{r}{K}P^2(t) \tag{4}$$

C. Calculating the Maximum Growth Rate

The growth rate of a patch can be found directly.

$$\Delta P(t) = rP(t) - \frac{r}{K}P^2(t) \tag{5}$$

To obtain integer population changes for our atomic puck setting, we round $\Delta P(t)$ to the nearest integer.

To find the maximum growth rate in unit time we use the derivative of $\Delta P(t)$.

$$\Delta P(t)' = r - 2\frac{r}{K}P(t) \tag{6}$$

and solve for $\Delta P(t)' = 0$, obtaining $P(t) = \frac{K}{2}$ and thus $\max \Delta P(t) = \frac{rK}{4}$. An appealing feature of the Verhulst model is that the maximum population growth rate is simply $\frac{rK}{4}$ during the period s, or $\frac{rK}{4s}$ per unit time, and the most productive population size is half the carrying capacity.

An example plot of $\Delta P(t)$ against $P(t)$ with the same parameters as before is shown in Figure 3 illustrating that the best growth is obtained when $P(t) = \frac{K}{2}$ and $\Delta P(t) = 0$ in the continuous case (left). In the discrete version there exists a range of populations that produce the optimal growth rate.

With this knowledge we can design a robot control strategy to sustainably forage such populations.

IV. Multi-Robot Foraging System

We consider a multi-robot foraging system containing a single home location and several distinct spatially-separated patches containing abstract atomic units of resource called "pucks". Robots receive a fixed reward for delivering a puck to the home location. There are N robots and M patches.

Each patch is a square area and its pucks are initially distributed uniformly at random. The task of each robot is to individually drive to a patch, find and collect a single puck and transport it to home. The patch puck population grows according to the logistic model, so that an integer number of pucks is created every period of s seconds. New pucks are placed at random in the square patch. A robot works for h_p seconds of "handling time" to pick up a puck in the patch and works h_d seconds to drop a puck at the home. After dropping a puck at home, a robot may be inactive or "sleep" for some time before returning to collect another puck. The sleep time enables the fine control of robot work time in a patch.

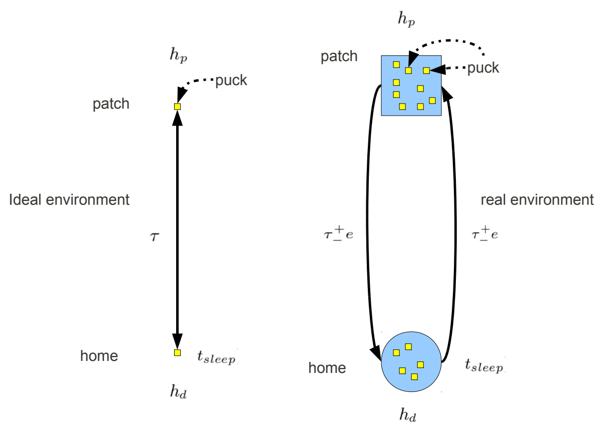


Fig. 4: Idealized model (left), where pucks appear at a single point, giving a constant travel time τ between home and patch. In the more realistic model (right) in which pucks are distributed around the patch, the travel time τ_-^+e varies slightly each due to the slightly different locations of puck collection and delivery. We assume that handling times h_p and h_d are constant in both models.

Robots move at a constant speed when driving between a patch and home, so each patch has an associated travel time τ that characterizes its distance from the home location.

For simplicity we assume there is no spatial interference between the robots, but we recognize this is an important problem for real robot systems and save this for future work. Although here each robot can carry only a single puck, the method is trivially extended to arbitrary carrying capacities.

For any patch, we define the Round Trip Time d:

$$d = \tau + h_p + \tau + h_d + t_{sleep} \tag{7}$$

which is the time required to complete a whole work cycle of driving from home to patch, collecting a puck, returning home, dropping the puck and (possibly zero) sleeping.

A. Foraging system model

We consider two models of the foraging system: an idealized model where all parameters including d are constant, and an arguably more realistic model where d can vary slightly with each work cycle. The idealized model is shown in Figure 4 (left). We ignore the distribution of pucks in the patch, and assume all pucks are found at the mean puck location - the center of the patch. Similarly, we assume pucks are dropped exactly at the home point. This gives us a constant drive time τ between puck collection and delivery. In the more realistic model of Figure 4 (right) pucks are collected and delivered at random locations around the patch and home centers. In the idealized model the drive time τ between collection and delivery is constant, while it varies slightly in the more realistic model, approaching τ on average over multiple work cycles.

B. The problem

The problem is to allocate an amount of robot work time to a patch such that the Maximum Sustainable Yield is achieved. This results in the maximum sustainable rate of pucks delivered to home. We assume that an unlimited number of robots are available. The general strategy is to allocate robot time to a patch by calculating the ideal, possibly non-integer, number of robots required, rounding up to the nearest larger integer number of robots, then adjusting the allocated robots' sleep time to achieve exactly the desired work rate (the same effect could be achieved by varying the robots' drive speed).

V. Static optimal work allocation to a patch

As a first approach, we can calculate an exact optimal robot work allocation to a patch assuming a constant τ. Combining the logistic model with the robot foraging model we obtain the maximum sustainable yield when the equality holds

$$\frac{rK}{4}s^{-1} = \frac{N_j}{h_p + 2\tau_j + h_d} \tag{8}$$

where the left hand side is the maximum increase in patch population size per unit time, and right hand side is the number of pucks harvested from patch j in unit time by N_j robots. Rearranging, we obtain the the optimal (continuous valued) number of robots by

$$N_j = (h_p + 2\tau_j + h_d)\frac{rK}{4}s^{-1} \tag{9}$$

Since we must allocate whole numbers of robots to patches, we choose the next larger integer number of robots $\lceil N_j \rceil$ and have each robot refrain from working for a short time each cycle to achieve effectively the exact optimal allocation N_j. The optimal duration of this non-working period - the "sleep time" - is:

$$t_{sleep} = \lceil N_j \rceil s\frac{4}{rK} - h_p - 2\tau_j - h_d \tag{10}$$

A. Results in the two models

The performance of the fixed "optimal" work allocation on the fixed and variable foraging models model is shown in Figure 5, for carrying capacity $K = 100$ and characteristic growth rate $r = 0.4$. A single patch exists, with an optimal allocation $N_j = 1.5$ robots. Results are shown for each of $N = 1, 1.5, 2$ robots. Consider first the idealized model: Figure 5 (a) shows the patch population size over time, while (b) shows the rate of puck delivery to home, for each value of N. When $N = 1$ the population climbs to reach around 80 pucks and the puck delivery rate starts at 10 and falls to 6 or 7 pucks per unit time. The patch is being under-harvested. For $N = 2$ the population and puck delivery rates both quickly fall to zero: the patch has been over-harvested and is permanently destroyed. As expected, when $N = 1.5$, the patch size is maintained at $\frac{K}{2} = 50$, giving a constant production (and thus delivery) rate of 10 pucks per unit time.

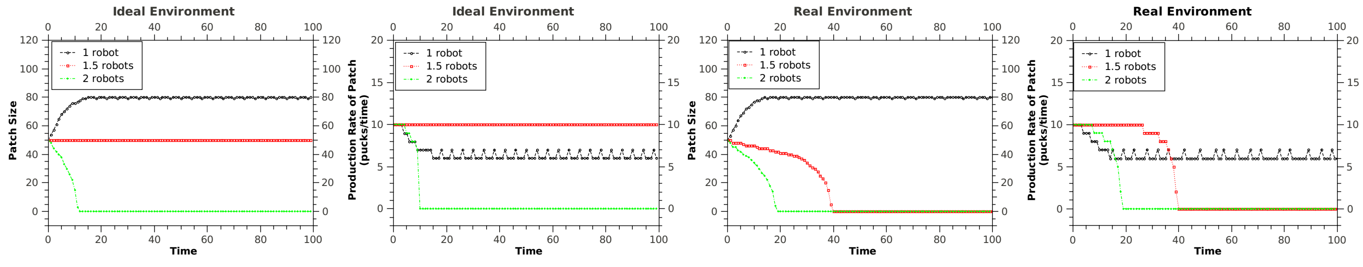
Now consider the variable model, where τ is only *approximately* constant. The results in Figure 5 (c) and (d) show that the behaviour for $N = 1$ is similar, and when $N = 2$ the patch is also quickly destroyed. Importantly, we see that a fixed allocation of $N = 1.5$, i.e. two robots, each sleeping for 25% of their time, also destroys the patch in finite time: the strategy does not produce sustainable harvesting in the presence of even a small variation in model parameters. This is due to a positive feedback effect: if the assumed value of τ is slightly different to the the actual experienced value for a single cycle, then the robot sleeps for too short or long a time and the patch is slightly over- or under-exploited locally. This reduces the productivity of the population, driving it away from the optimal. Since this occurs whether the error in τ is positive or negative, the random variations do not cancel out and the system is driven either to under-exploitation or patch destruction.

The sensitivity of the MSY approach is well known, and is a problem when parameters are actually variable, when measurements of them have some error, and when some foragers "cheat" by taking more than their allocation. This is a source of criticism of the method being used historically to "optimize" catch quotas in real-world fisheries, with subsequent collapses of fish populations. Small errors in parameter estimates and variations in the harvested amounts can quickly lead to destruction of the population. It may be too risky to manage critical natural resources using this method alone [15], [16].

In the remainder of the paper, we extend the basic model to (i) handle multiple patches; and (ii) dynamically adapt the individual robot work rate at run time to prevent the runaway feedback effect from occuring and maintain the foraged population at the optimal level.

B. Allocating robots to patches

In the general case we have multiple patches to forage with our pool of robots, so we need to allocate robots to patches. There are two possible cases to consider; one in which the number of robots is less than or equal to the minimum required optimally forage all patches, and another when there are more than enough robots to do the work so we can over-allocate (the redundancy providing robustness to robot failures). Excess work capacity is absorbed by "sleep time" in which the robot stops working for a short period. Equivalently the robot could work continuously but at a slower rate.



(a) Patch population (b) Production Rate (c) Patch population (d) Production Rate

Fig. 5: The results of fixed-allocation foraging in fixed (a,b) and variable (c,d) scenarios. The optimal robot allocation is $N_j = 1.5$ robots. Results for $N = 1, 1.5, 2$ are shown.

a) Case 1: No redundant robots: When $N \leq \sum_{j=1}^{M} \lceil N_j \rceil$ we allocate robots to patches in order of increasing round-trip time d_j in order to maximize the overall puck delivery rate. Let the ordered list $L = \{N_1, N_2, \ldots, N_M\}$ contain the optimal static robot allocations for all M patches, calculated using Equation 9. The list is sorted by increasing patch round trip time d_j i.e. $d_i \leqslant d_j$ for any $i < j$. $\hat{N}_j = \lceil N_j \rceil$ robots are allocated to the jth patch in this list in order, until the pool of robots is used up.

b) Case 2: Redundant robots: If there are more than enough robots to forage all patches optimally, i.e. $N > \sum_{j=1}^{M} \lceil N_j \rceil$, then we over-allocate by distributing all N robots proportional to the needs of each patch, allocating integer $\hat{N}_j$ robots to patch P_j thus:

$$\hat{N}_j = \begin{cases} round(N \cdot \frac{\lceil N_j \rceil}{\sum_{j=1}^{M} \lceil N_j \rceil}) & \text{if } j = 1, 2, \ldots, M-1 \\ N - \sum_{j=1}^{M-1} \hat{N}_j & \text{if } j = M \end{cases} \tag{11}$$

C. Initializing Sleep Time

The sleep time for a robot foraging patch P_j is chosen to absorb the excess work capacity of the allocation of robots $\hat{N}_j$ (if any) to closely approximate the optimal allocation N_j. This is the same value for every robot on the patch:

$$t_{sleep} = \begin{cases} 0 & \text{if } \hat{N}_j \leqslant N_j \\ \hat{N}_j \cdot s \cdot \frac{4}{rK} - h_p - 2\tau_j - h_d & \text{if } \hat{N}_j > N_j \end{cases} \tag{12}$$

D. Adaptive Sleep time

As discussed above, a fixed work allocation will inevitably drive the patch population away from the optimal in the presence of variation or measurement errors. To address this issue, we propose a simple method of adapting overall work rate by having each robot locally modify its sleep time in response to its own recently-sensed estimate of patch population $p_{obs}(t)$. On each work cycle, the robot adjusts its sleep time as follows, sleeping longer if the population is

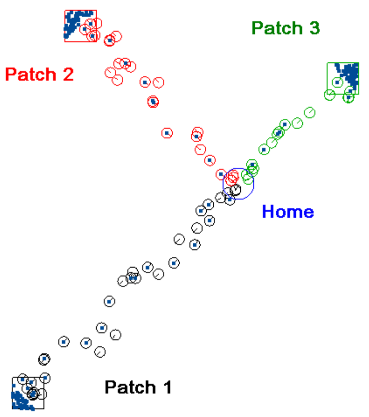


Fig. 6: Screenshot from the Antix simulator. 80 robots (small circles) adaptively forage pucks (dark dots) from 3 patches (squares) and deliver them to the home (large circle).

observed to be too small, or shorter if the population is too large:

$$p_{error}(t) = \frac{K}{2} - p_{obs}(t) \tag{13}$$

$$\Delta t_{sleep}(t) = \begin{cases} k_{incr} \cdot p_{error}(t) & p_{error} > 0 \\ k_{decr} \cdot p_{error}(t) & p_{error} < 0 \\ 0 & p_{error} = 0 \end{cases} \tag{14}$$

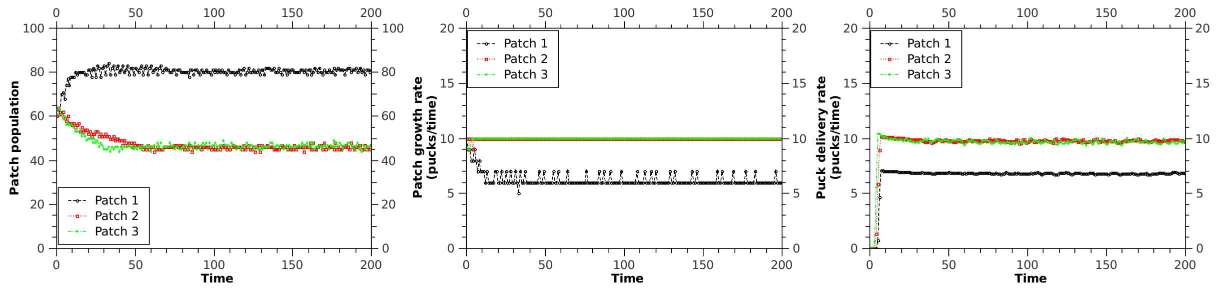
where k_{incr} and k_{decr} are gain parameters which control the frequency response of this proportional controller.

This algorithm is extremely simple, but solves the problem effectively as shown below.

E. Adaptive Multi-Patch Foraging Demonstration

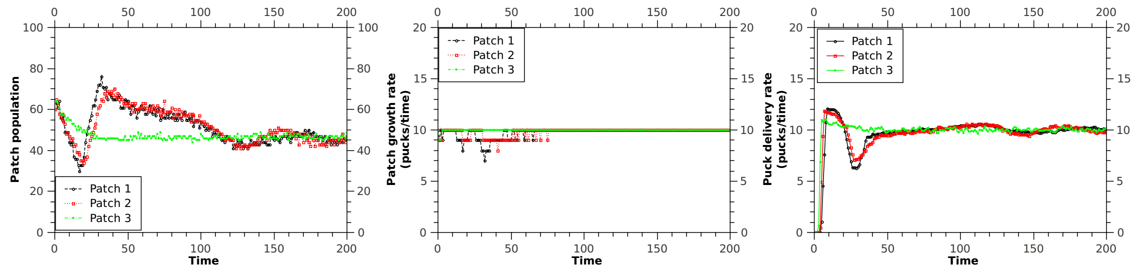
We demonstrate the adaptive controller in the freely-available sensor-based robot simulator Antix[2]. Pucks are placed at random in the patches, the robot drives between home and goal using a simple kinematic controller, and detects pucks using an on-board sensor with limited range. Thus the simulation approximates the foraging model with variable τ. We use a simple robot simulator rather than the

[2]http://github.com/rtv/Antix



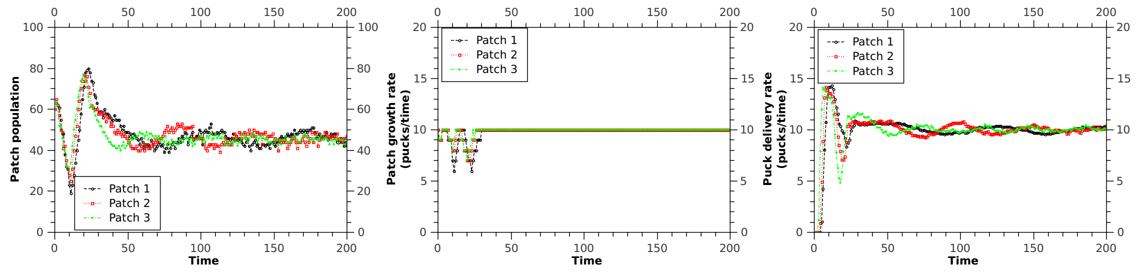
(a) Patch population (b) Patch growth rate (c) Puck delivery rate

Fig. 7: Results: 60 robots adaptively foraging 3 patches of randomly placed pucks. Allocation is patch 1:=18 robots, 2:=24, 3:=18. Sleep time is zero. Sustainable, but too few robots to maximize yields of all three patches. Time is measured in hundreds of seconds. Puck delivery rate is patch 1:=6.81 pucks/time, 2:9.69, 3:=9.77, all:=26.27.



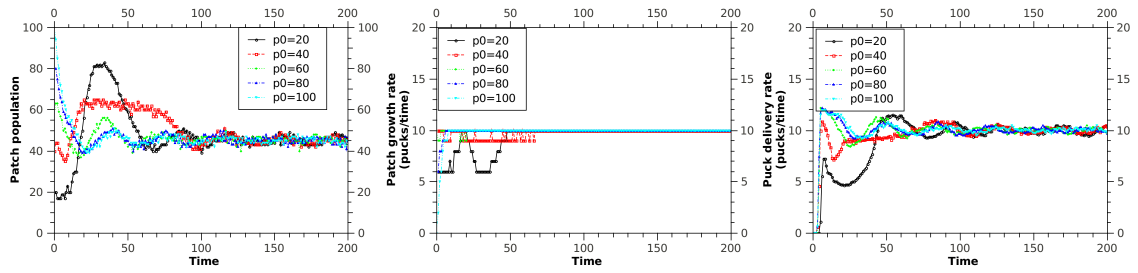
(a) Patch population (b) Patch growth rate (c) Puck delivery rate

Fig. 8: Results: 80 robots adaptively foraging 3 patches of randomly-placed pucks. Allocation is patch 1:=35 robots, 2:=27, 3:=18. Sustainable and optimal (maximum productivity is 10 pucks/unit time).



(a) Patch population (b) Patch growth rate (c) Puck delivery rate

Fig. 9: Results: 100 robots adaptively foraging 3 patches of randomly placed pucks. Allocation is patch 1:=43 robots, 2:=33, 3:=24. Sustainable and optimal (maximum productivity is 10 pucks/unit time).



(a) Patch population (b) Patch growth rate (c) Puck delivery rate

Fig. 10: 20 robots forage on patch 3 only. The plots show the system evolution for different starting patch sizes {20,40,60,80,100}. The system reaches optimal sustainable foraging in each case.

more abstract model above to demonstrate the robustness of the adaptive method. We disable robot-robot collisions in the simulator to prevent spatial interference effects from dominating the results of this first study. New pucks are placed at random in their patch, so small variations of transportation time are present in the simulation.

As before, the puck growth rate is $r = 0.4$ and carrying capacity $K = 100$. Initial population P_0 is 60 pucks. Puck handling times are $h_p = 10$ seconds for and $h_d = 8$ seconds for drop off. The simulation runs for 20,000 simulated seconds. The patch growth period is $s = 100$ seconds. The adaptive controller gains are $k_{incr} = 4.0$ and $k_{decr} = 3.5$. The robot start times are staggered to avoid all the robots reaching patch and home at same time.

There are three patches, all with the same logistic growth parameters, but located at different distances from home, at 2, 3 and 4 times unit distance. We run the system with robot populations of 60, 80 and 100 robots.

The overall performance metric we mean to optimize is the sustained delivery rate of the entire robot system, which is simply the total number of pucks delivered by all robots per unit time. With these parameters the optimal delivery rate is 10 pucks per unit time.

Figure 7 shows the system evolution for 60 robots, where plot (a) shows the patch population over time. The population of patches 2 and 3 is decreased to the optimal population size of 50 after around 50 * 100 seconds. Patch 1 has the largest round trip time, and not enough robots were available to service it completely. Its population grows to the carry capacity, which drops the growth rate (b) to below the maximum of 10 pucks per unit time. The number of robots allocated to each patch is shown in the caption. The foraging is sustainable and optimal on patches 2 and 3 but trivially sustainable and suboptimal on patch 1 due to lack of robot work capacity.

Figure 8 shows the system evolution for 80 robots. The patch population plot (a) shows the robots initially over-harvest all patches and the populations drop quickly. Adapting to the falling population, the robots increase their sleep time and the population climbs again, overshooting the ideal size until the robots adapt again, bringing the population back to the approximately optimal size. The patch growth rate is shown in (b) and an early drop can be seen until recovery at around $80 * 100$ seconds. The puck delivery rate is seen in (c), climbing from zero as robots are are deployed, rising above 10 pucks per unit time as the patch is over-harvested, dropping as the population declines, then converging to vary slightly around the optimal of 10 pucks per unit time for each patch.

Figure 9 shows the results for 100 robots. A similar evolution as before can be seen, but with larger value swings over shorter time. Again the system converges to approximately the Maximum Sustainable Yield. The excess work capacity has been turned into sleep time to avoid over-harvesting.

To demonstrate the controller's ability to drive the system into MSY from different initial conditions, we had 20 robots forage on patch 3 only, and repeat for different initial patch sizes $\{20, 40, 60, 80, 100\}$. Figure 10 shows the system evolution, which converges to the MSY in each case.

VI. Conclusion and Future Work

This paper introduced the Sustainable Robot Foraging problem and describes how the MSY model can be applied to obtain a practical and near-optimal task-allocation for a multi-robot system. Online fine-grained feeback control of robot work rate is required to achieve maximum sustainable yield due to variations and/or uncertainties in all practical systems.

The method presented here does fine-grained adaptation to an initially close-to-optimal global robot allocation. The initial allocation is easily obtained if the logistic model parameters for each patch is known, and no communication between robots is required for convergence. In future we will present another controller that converges (usually more slowly) to the optimal behaviour even when no model parameters are initially known, using only locally-obtained information and modest local communication between robots.

Also of interest is a separate mechanism to robustly avoid the destruction of patches. This is easily done if the minimum viable population is known, but when this value is not available we may simply set a conservative threshold below which we stop harvesting. If we have no a priori data on how to set the threshold, we may be able to estimate it by observing the productivity / production curves by sampling, or in the worst case we may observe the accidental destruction of patches and adapt to avoid repeating the same mistake.

Finally, recognizing that spatial interference can dominate performance in high-density robot populations, we should examine its effects on sustainable foraging. One intriguing possibility is that interference may make pucks harder to find when patch population is small and a patch is full of robots, reducing harvesting rates and allowing the patch to regrow. This is a potentially useful source of emergent feedback control to avoid destroying a patch[3].

This is an early step towards the development of machines that can harvest biomass from the environment indefinitely without damaging it. This is a challenge that has defeated even the smartest primates, historically.

Open Experimental methodology

All source code, scripts and configurations used to generate the results in this paper are available for download at `http://autonomylab.org/pub/song_IROS2013.tar.gz`. The SHA1 hash of the package begins `ec85ab8febe0e9d7`.

Acknowledgements

This work was supported by NSERC in Canada.

[3]This idea is due to Jens Wawerla.

References

[1] P. Verhulst, “Notice sur la loi que la population suit dans son acroissement,” *Corr. Math. et Phys*, vol. 10, p. 113, 1838.

[2] T. Malthus, “An essay on the principle of population,” *London*, 1798.

[3] J. Hjort, G. Jahn, and P. Ottestad, “The optimum catch,” *Hvalradets Skrifter*, vol. 7, pp. 92–127, 1933.

[4] L.-A. G. Etienne Danchin and F. Czilly, “Behavioural ecology,” *Oxford University Press*, 2008.

[5] D. W. Stephens, J. S. Brown, and R. C. Ydenberg, *Foraging*. Chicago: University of Chicago Press, 2007.

[6] J. Wawerla and R. T. Vaughan, “Online robot task switching under diminishing returns,” *In Proceeding of the Twelfth International Conference on Aritifical Life (ALife XII)*, pp. 789–796, 2010.

[7] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng, “Cooperative mobile robotics: Antecedents and directions,” *Autonomous Robots*, vol. 4, pp. 7–27, 1997.

[8] K. Lerman and G. A., “Macroscopic analysis of adaptive task allocation in robots,” *In Proc of the Int. Conf. on Intelligent Robots and Systems (IROS-2003), Las Vegas, NV*, pp. 1951–1956, 2003.

[9] M. Ashikaga, M. Kikuchi, T. Hiraguchi, M. Sakura, H. Anonuma, and J. Ota, “Foraging task of multiple mobile robots in a dynamic environment using adaptive behavior in crickets,” *Journal of Robotics and Mechatronics*, vol. 19, no. 4, pp. 446–473, 2007.

[10] R. Pearl and L. Reed, “On the rate of growth of the population of the united states since 1790 and its mathematical representation,” *Proc. Nat. Acad. Sci.*, vol. 6, pp. 275–288, 1920.

[11] F. E. Smith, “Population dynamics in dapthnia magna and a new model for population growth,” *Ecology*, vol. 44, no. 4, pp. 651–663, 1963.

[12] H. Fujikawa, A. Kai, and S. Morozumi, “A new logistic model for escherichia coli growth at constant and dynamic temperatures,” *Elsevier Ltd.*, vol. Food Microbiology 21, pp. 501–509, 2004.

[13] E. Yoshida, T. Arai, J. Ota, and T. Miki, “Effect of grouping in local communication system of multiple mobile robots,” *Intelligent Robots and Systems ’94. ’Advanced Robotic Systems and the Real World’, IROS ’94. Proceedings of the IEEE/RSJ/GI International Conference on*, vol. 2, pp. 808 –815 vol.2, sep 1994.

[14] V. Tereshko and A. Loengarov, “Collective decision-making in honey bee foraging dynamics,” *School of Computing, University of Paisley*, pp. 1–7, 2005.

[15] C. Clark, “The economics of overexploitation,” *Science*, vol. 118, pp. 630–634, 1973.

[16] A. Punt and A. Smith, “The gospel of maximum sustainable yield in fisheries management: birth, crucifixion and reincarnation,” in *Conservation of exploited species*, ser. Conservation biology series, J. Reynolds, Ed. Cambridge University Press, 2001. [Online]. Available: http://books.google.ca/books?id=W8WldjwSjZYC