Optimal Online Data Sampling or How to Hire the Best Secretaries

Yogesh Girdhar and Gregory Dudek
McGill University
Montreal, QC, Canada
{yogesh,dudek}@cim.mcgill.ca

Abstract

The problem of online sampling of data, can be seen as a generalization of the classical secretary problem. The goal is to maximize the probability of picking the k highest scoring samples in our data, making the decision to select or reject a sample on-line. We present a new and simple on-line algorithm to optimally make this selection. We then apply this algorithm to a sequence of images taken by a mobile robot, with the goal of identifying the most interesting and informative images.

1. Introduction

This paper considers how to identify good data samples with a sensor, for example how to select landmarks with a roving robot as it sees a series of images passing before it. More pragmatically, if a robot system is reconstructing a scalar or vector field over a sampling domain, how can it select the best small set of samples?

Consider a game in which you are presented with a finite sequence of random numbers and then your goal is to identify the maximum number in this sequence. You are only allowed to make one selection, and you cannot go back and select a previously rejected number. There is no statistical information about the sequence available other than the size of the sequence. This problem is popularly known as the secretary problem, since the task can be posed as hiring the best secretary, with the interview score of the candidates represented as a sequence of random numbers. If we are allowed to go back to a previously rejected sample, then our algorithm is trivial. We can just go through all the samples, identify the maximum value, and then in the end go back and choose the sample with the maximum value. Hence, the main twist in the problem arises due to the fact that when a sample is presented we must make a decision to select it or not immediately.

Of course, the problem of hiring a secretary serves as a metaphor for any generic problem where one must perform on-line data selection without revision: that is, one must select a sample from a stream of data without being able to revise the decision. In particular, in the context of vision, and specifically robotic vision, this is the problem we face if a vehicle must select an image, or a set of measurements, as it executes a sampling trajectory. The problem of making an irrevocable decision to choose a sample relates to an idealized problem, but it has many related analogues based on bandwidth limitations, decision making or other factors.

In this paper we will discuss several variations on the problem and pose an algorithmic solution. We then validate it via a simple image acquisition task. Our main focus is however on the problem of choosing a known number of samples k such that the probability of them being the top k samples in the entire sequence is maximal.

This problem definition and the related analysis and solution strategies have application in a wide variety of fields beyond vision and robotics, such as online auctions and business resource allocation.

Consider the problem of having a robot estimate some function of a 2-D landscape by deploying a fixed number of static sensors. We would then like the robot to drop the sensors at locations which give us maximum information. Since a robot path is a 1-D line, we then discretize this path and model the measurement at each point as a sample. If we do not assume any statistical distribution for these measurements(which is a safe assumption if our discretization is coarse), we can then apply our multi-choice secretary algorithm to identify the locations at which to drop the sensors.

To illustrate the effectiveness of our algorithm, we apply our technique to the selection of images in a context where the scoring function is an entropy-based measure used to pick out the most informative images.

Another possible application could be to the Vacation

978-0-7695-3651-4/09 $25.00 © 2009 IEEE
DOI 10.1109/CRV.2009.30

Snapshot Problem [4]. Imagine that you are on vacation and have only k photos remaining in your camera. You know that you will be visiting $n > k$ more interesting places in the coming few days. How can you devise a strategy that will maximize the probability of you getting photos of the most interesting places from your trip? With an appropriate *interestingness detector*, this could again be a direct application of the secretaries problem.

2. The Secretary Problem

The secretary problem has a long and varied history. Due to its broad relevance to different domains, it has been considered by different authors in several different contexts. It is generally accepted that Dynkin [6] was the first one to solve the problem formally. For an interesting discussion on the origins of this problem see [7]. Here is a description of the problem in its simplest form with $k = 1$:

- You are given the task of hiring the best possible candidate for the job of a secretary.
- There are n applicants who have signed up for the interview.
- After interviewing each candidate you can rank them relative to all other candidates seen so far.
- You must either hire or reject the candidate immediately after the interview.
- You are not allowed to go back and hire a a previously rejected candidate.

A typical strategy would be to just observe the first r candidates without accepting any, then find the highest score among them, and then hire the first candidate with score higher than that. This is known to be the optimal strategy for this problem. The problem now is to select the best value for r.

Let $\Phi(r)$ be the probability of successfully finding the highest scoring candidate, when we set the training interval to be r. We can then write $\Phi(r)$ as:

$$\Phi(r) = \sum_{i=r+1}^{n} P(S_i) \quad (1)$$

where S_i is the event that the ith candidate is the highest scoring candidate, and that our algorithm did not select any of the previous candidates. Hence we have:

$$\Phi(r) = \sum_{i=r+1}^{n} \frac{1}{n} \cdot \frac{r}{i-1} \quad (2)$$

$$= \frac{r}{n} \sum_{i=r}^{n-1} \frac{1}{i} \quad (3)$$

Here $1/n$ is the probability that the ith candidate is the highest scoring one, and $r/(i-1)$ is the probability none of the previous candidates were selected. $\Phi(r)$ can be optimized by optimizing the integral for which the above sum is the Riemann approximation. Doing that we find that $\Phi(r)$ is maximum when $r = n/e$, as $n \to \infty$. A detailed proof of this can be found in [5].

This problem can be generalized by allowing one to choose several samples instead of just one. Several different variants of the problem can then be posed. For example, we can choose to maximize the expected sum of the scores [9] [3] [2] , or maximize the probability that one of the selected samples is the highest scoring one [10], or maximize the probability that all k of the chosen samples are in fact the top k highest scoring ones, and so on. If we assume that each incoming random sample has different weights and values associated with it, then this problem becomes the knapsack secretary problem [1].

There are many other variants of this problem which exist in the literature. Freeman [8] reviews several of these variants.

3. The Multiple Choice Secretary Problem

Although the single secretary problem is fairly well understood, the problem of hiring multiple secretaries (or the problem of selecting a finite set of data samples) has several variations and a less definitive algorithmic methodology for a given problem domain.

For the case when the number of positions which need to be filled is more than one ($k > 1$), there are several possible ways in which the above single secretary solution can be generalized. In this paper we mainly focus on techniques to optimize the selection of all top k highest scoring candidates.

3.1. Previous Work

Kleinberg [9] suggested an algorithm to maximize the expected sum of the scores of the candidates. The algorithm works by splitting the candidates in two roughly half intervals chosen randomly using a binomial distribution $B(n, 1/2)$. We then recursively apply the classic($k = 1$) secretary algorithm to the first half of the candidates, choosing $l = \lfloor k/2 \rfloor$ candidates. While doing this we also find the lth highest scoring candidate from the first half and use this as a fixed threshold to select the remaining candidates in the second half.

Babaioff et al. [3], [2] suggest a simpler algorithm with the same goal of maximizing the expected sum of the scores of the selected candidates. The algorithm uses a sliding threshold to choose the candidates in the following way:

Algorithm: MaxExpectedScoreSum$(\{x_1, .., x_n\}, k)$

1. Find the top k scores in the first $r = \lfloor n/e \rfloor$ candidates, without selecting any. Call this list of thresholds, $T = \{t_1, .., t_k\}$.
2. When a candidate with score higher than the minimum score in T is encountered:
   (a) Hire the candidate
   (b) Remove the minimum score value from the set T.
3. Continue with step 2 until the set T is empty, or all positions have been filled, or all candidates have been reviewed.

Note that both the algorithms above aim to maximize the actual numerical score of candidates. This is different from our goal of maximizing the probability of selecting the *all* k highest scoring candidates. The algorithm we present in this paper addresses the case when *all* of the top secretaries are selected. It does not distinguish between the failed cases of maybe only being able to select the top $k-1$ secretaries or no top ranking secretaries at all. The algorithm does not try to maximize the expected sum of the scores.

4. Selecting Top-k Secretaries Using a Fixed Threshold

We now present an approach where we use a single threshold to optimally select the top k highest scoring candidates. The threshold is chosen as the maximum observed score in the first r candidates. We would like r to be a function of k such that if k increases, then r decreases. We compute r by maximizing the probability of success $\Phi(r)$, where success is defined by the event that all of the top k highest scoring candidates have been selected.

Let S_i^k be the event that with the selection of the ith candidate, we have succeeded. We can then write :

$$\Phi^k(r) = P(Success) \quad (4)$$
$$= P(S_1 \cup \ldots \cup S_n) \quad (5)$$
$$= P(S_{r+k} \cup \ldots \cup S_n) \quad (6)$$

We can ignore the first r candidates since those candidates are never selected as per our algorithm definition, and then we can ignore the next $k-1$ candidates since its impossible to select k candidates from $k-1$ possibilities. Analogous to Equation 2, we can then write $\Phi^k(r)$ as:

$$\Phi^k(r) = \sum_{i=r+k}^{n} P(S_i^k) \quad (7)$$

$$= \sum_{i=r+k}^{n} \frac{k}{n} \cdot \frac{r}{i-1} \cdot \frac{\binom{i-r-1}{k-1}}{\binom{n}{k-1}} \quad (8)$$

$$= \frac{k}{n} \cdot \frac{r}{\binom{n}{k-1}} \cdot \sum_{i=r+k+1}^{n-1} \frac{\binom{i-r}{k-1}}{i} \quad (9)$$

Lets look at the three components of Equation 8. The first term: k/n is the probability that the ith candidate is one of the top k candidates. The second term: $r/(i-1)$ is the probability that none of the previous candidates were the last of the top k selected candidates. These two terms are similar to the two terms in Equation 2. The third term: $\binom{i-r-1}{k-1} / \binom{n}{k-1}$ is the probability that all of the remaining $k-1$ candidates have been selected. Combining these terms we get the probability of the event that we selected the ith candidate, and with that we have successfully selected all top k candidates.

Figure 1 shows a plot of $\Phi^k(r)$ for different values of k.

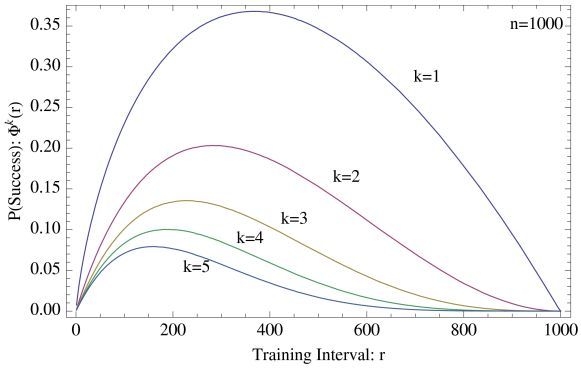

Figure 1. Probability of success in selecting all top-k highest scoring candidates as a function of the training interval r, for fixed threshold solution.

4.1. Optimal Training Interval

Let $\hat{r}$ be the values of r for which the probability of success $\Phi^k(r)$ is maximum. Numerically finding the maximas for $n = 1000$, and $k = 1..20$, and then fitting a curve to these points, we found an expression for $\hat{r}$ as $n \to \infty$, and $k \to \infty$:

$$\hat{r}(k) = \frac{n}{ke^{1/k}} \quad (10)$$

Figure 2 shows a plot of this function along with the actual data points used for fittings for . Simplicity of this expression and accuracy of its fit with the data gives us a hint

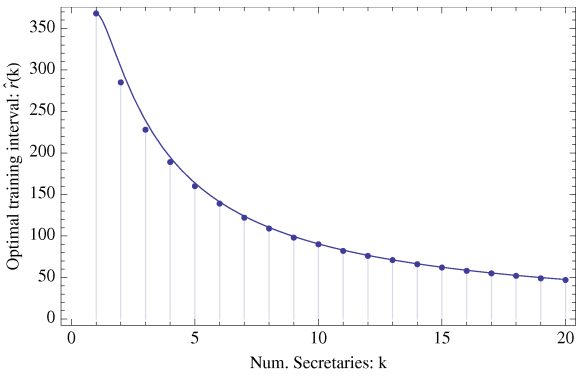

Figure 2. Optimal training interval as a function of number of secretaries k, for fixed threshold solution.

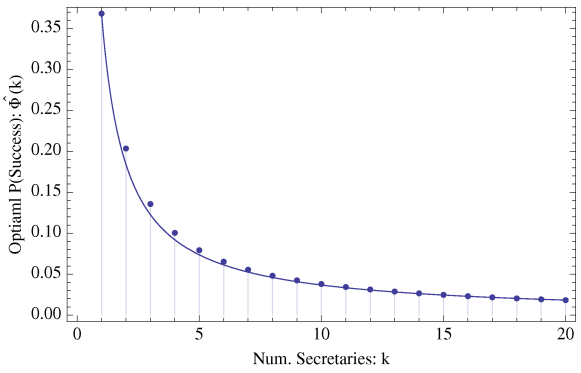

Figure 3. Optimal probability of success as a function of number of secretaries k, for the fixed threshold solution.

towards its correctness. Due to lack of time, a formal proof was not attempted in this paper.

With an expression for the optimal training interval $\hat{r}(k)$, we can now summarize our algorithm. Let x_i be the score of the ith candidate.

Algorithm: Top-k($\{x_1, .., x_n\}$, k)

1. Let $r = \lfloor \frac{n}{ke^{1/k}} \rfloor$ be the observation or training interval.
2. Find threshold $t = Max(\{x_1, .., x_r\})$.
3. Select each candidate from the interval $\{r+1, .., n\}$ with score higher than the threshold till we select k candidates or we run out of candidates.

4.2. Probability of Success

In the algorithm above, we are maximizing the probability of successfully finding out top-k candidates. We now try to find an expression for this. Let $\hat{\Phi}(k)$ be the maximal probability of success for a given k value. We then have:

$$\hat{\Phi}(k) = \Phi^k(\hat{r}(k)) \tag{11}$$

We again numerically compute $\Phi^k(\hat{r}(k))$ for $k = 1..10$ and fit a curve to these points to get an expression for $\hat{\Phi}(k)$ as $n \to \infty$, and $k \to \infty$:

$$\hat{\Phi}(k) = \frac{1}{ek} \tag{12}$$

Figure 3 shows a plot of this function along with the data points used for fitting.

With this algorithm in hand, we are able to select data samples, and in particular images, from a data stream observed by a camera or robot. In the next section we examine the experimental performance of this approach.

5. Experiments

5.1. Simulations

In the following section we compare the performance of our fixed threshold algorithm, with the Babaioff et. al.'s sliding threshold algorithm. Our judging criterion is the probability of selecting the top k highest scoring candidates.

We use a simulation such that for $n = 100$ candidates, we generate a random permutation of score ranks and then run each algorithm with this data set. For each $k = \{1, .., 10\}$, we run the simulation 10000 times to get an expected probability of success.

Figure 4 shows the results of this simulation in terms of the actual probability of success in selecting the k highest-scoring candidates. These results indicate a good agreement between our theoretical predications and the empirical results of the method. In addition, it confirms that the fixed threshold algorithm has superior performance. Clearly, as the number of samples being selected from a finite universe of data increases, the problem finding precisely the optimum samples becomes more and more difficult.

5.2. Optimal Image Data Sampling

We are particularly interested in the problem of selecting sample images acquired from a robot, such that it can automatically identify the most informative locations in the world, for the purpose of placing static sensor nodes. To explore this idea further, we assume to have a robot (such as AQUA [11]), equipped with a camera and a set of k deployable static sensor nodes. As the robot moves around in the world, it takes visual measurements once every few seconds, and then gives each measurement a score. The goal then is to identify the highest scoring sites online, so that the sensors can be deployed.

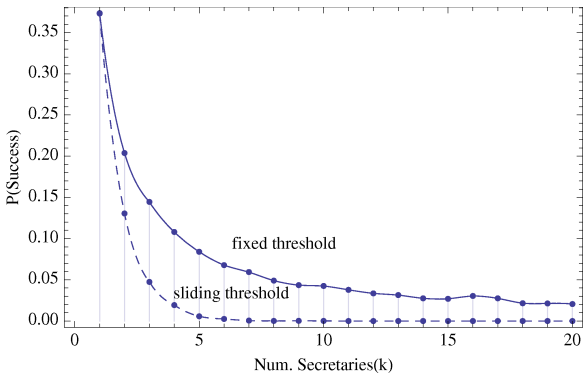

Figure 4. Result of simulation comparing probability of successfully choosing all of the top-k highest scoring candidates out of a total of $n = 100$. Not that the curve corresponding to the fixed threshold algorithm (solid line) is same as Figure 3. We see that our fixed threshold algorithm performs consistently better.

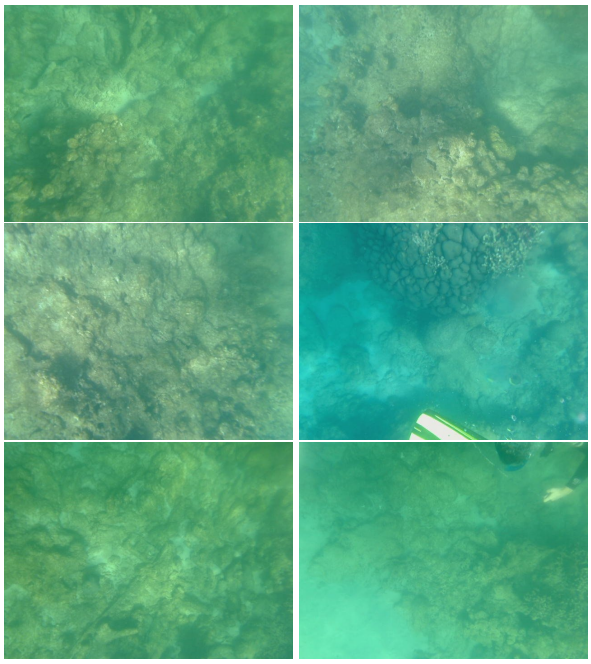

Figure 5. Coral Reef Data-set. Six randomly chosen images from the total of 336 images in the coral reef data set, along with the histogram and scatter plot of the scores.

For the purpose of this experiment, we use a very simple pixel entropy based measurement score. The score of a measurement image was defined as sum of pixel entropy for each of its RGB channels. This simple metric should in principal favor images with more colors than those which don't have as many colors. To do this, for each color channel, we computed an intensity histogram with 32 bins, and then computed its total entropy.

We used a data set consisting of 336 images of a coral reef, collected by swimming on top of it with a camera pointing downwards. The images have a mean score of 9.00687, and variance of 1.57866. Figure 5 shows a few images from our data set, and Figure 6 and 7 show the scatter plot and histogram of their scores.

We ran our fixed threshold algorithm with $k = 6$, so that we get six or less images. Our algorithm only managed to return two images, however these two images were the two highest scoring images in our data set with score (pixel entropy) of 12.4692 and 12.4398 respectively. The optimal solution in this case is the five images with scores $\{11.72, 12.06, 12.20, 12.32, 12.44, 12.47\}$ (see Figure 8).Our algorithm selects only two of the images because the third-highest scoring image happens to fall quite early in the sequence and hence is chosen as our threshold.

Inverting the sequence of images in our data set gives us a new data set without changing the temporal structure of the sequence. Hence, for the purpose of experimentation, we run our algorithm on the inverted sequence of images from the coral reef data set. This time we got six high scoring images (Figure 10), out of which three exist in the actual top six highest scoring images as seen in Figure 8.

Figure 9 shows these selected images. It is interesting to see that the algorithm picked out images of the swimmer as the highest scoring images.

6. Discussion

In this paper we have discussed the problem of online sampling, which can be seen as a generalization of the classical secretary problem. We have presented a new and simple problem definition and associated algorithm which optimally chooses the k highest scoring samples in our data set, online. As far as we know this is the first work which deals with this version of the generalized secretary problem.

Our work demonstrates the relevance and utility of the secretary problem to robotic sampling. As a sample application, we applied our fixed threshold algorithm to the vacation snapshot problem, producing a solution which works online. Although we used a very simple scoring function, this simple module can be replaced with more complex ones to get better results.

The secretary algorithms discussed in this paper are only optimal when there is no prior information about the scores, and there is no correlation between the scores of neighbor-

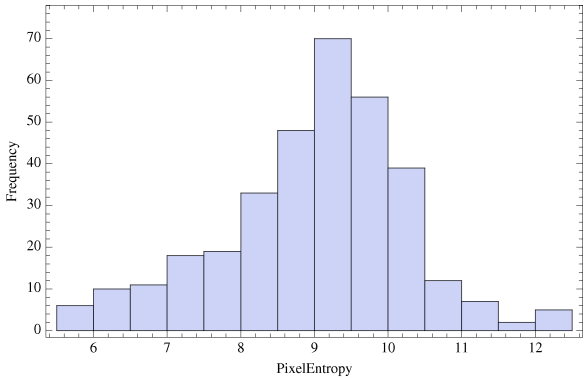

Figure 6. Histogram of the image scores in the coral reef data set.

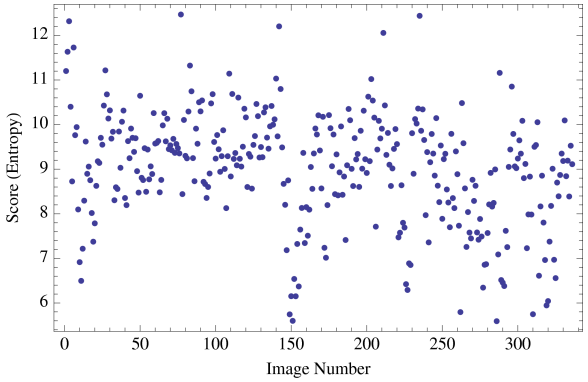

Figure 7. Scatter plot of image scores in the coral reef data set.

ing samples. For the problem of selecting sample images by a mobile robot, normally the neighboring samples will have some correlation. In practice these correlations are very hard to model or predict. For example in the presence of occlusion boundaries close to the camera, successive frames can be essentially uncorrelated. Our algorithm thus can be regraded as dealing with the worst case scenario.

In addition when the video sequence is subsample at a low frame rate, any existing correlation is likely to be drastically decreased. For the purpose of selecting images that give most amount of information about the world, we would ideally like our samples to not only be highly informative, but also be independent with no mutual information between each other. We hope to deal with these issues in our future work.

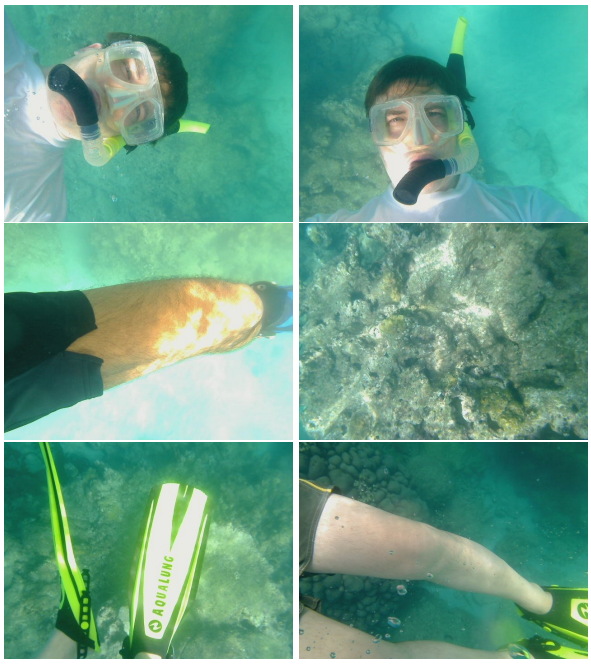

Figure 8. Top 6 highest scoring images in the coral reef data set, with scores of {11.72, 12.06, 12.20, 12.32, 12.44, 12.47 }

Acknowledgement:

The authors would like to thank Philippe Giguere for his help in discussing and developing the ideas presented in this paper. We also gratefully acknowledge the advice and assistance of Luc Devroye.

References

[1] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. A knapsack secretary problem with applications. In *APPROX '07/RANDOM '07: Proceedings of the 10th International Workshop on Approximation and the 11th International Workshop on Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 16–28, Berlin, Heidelberg, 2007. Springer-Verlag.

[2] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Online auctions and generalized secretary problems. *SIGecom Exch.*, 7(2):1–11, 2008.

[3] M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 434–443, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[4] E. Bourque and G. Dudek. Automated image-based mapping. In *IEEE Computer Vision and Pattern Recognition*

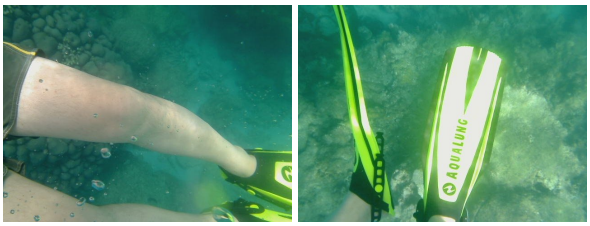

Figure 9. Highest scoring images chosen as a result of running the fixed threshold algorithm with $k = 6$. The corresponding scores are: $\{$12.4692, 12.4398$\}$. Since our scoring function is based on pixel entropy of the image, the algorithm skips over most of images of coral reef and instead picks more colorful *accidental* images of the swimmer.

(CVPR)–Workshop on Perception of Mobile Agents, pages 61–70, June 1998.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, USA, 2001.

[6] E. Dynkin. The optimum choice of the instant for stopping a markov process. *Soviet Math. Dokl*, 4, 1963.

[7] T. S. Ferguson. Who solved the secretary problem? *Statistical Science*, 4(3):282–296, 1989.

[8] P. .Freeman. The secretary problem and its extensions: A review. *International Statistical Review*, 1983.

[9] R. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 630–631, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[10] M. Sakaguchi. Dowry problems and ola policies. *Rep. Statist. Appl. Res. JUSE*, 1978.

[11] J. Sattar, G. Dudek, O. Chiu, I. Rekleitis, A. Mills, P. Gigure, N. Plamondon, C. Prahacs, Y. Girdhar, M. Nahon, and J.-P. Lobos. Enabling autonomous capabilities in underwater robotics. *"Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

Figure 10. Highest scoring images chosen as a result of running the fixed threshold algorithm with $k = 6$, on reverse sequence of images in the coral reef data set. The corresponding scores are: $\{$11.16, 12.44, 12.06, 11.02, 12.20, 11.03$\}$.