Linköping University | Department of Computer and Information Science
Master's thesis, 30 ECTS | Datateknik
2017 | LIU-IDA/LITH-EX-A--2017/001--SE

Learning to Search for Targets

– A Deep Reinforcement Learning Approach for Scanning Large Environments

Inlärd sökning efter mål

Oskar Lundin

Supervisor : Sourabh Balgi
Examiner : Jose M. Peña

External supervisor : Fredrik Bissmarck

LINKÖPING UNIVERSITY

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Upphovsrätt

Copyright

Abstract

The abstract resides in file `Abstract.tex`. Here you should write a short summary of your work.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque in massa suscipit, congue massa in, pharetra lacus. Donec nec felis tempor, suscipit metus molestie, consectetur orci. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Curabitur fermentum, augue non ullamcorper tempus, ex urna suscipit lorem, eu consectetur ligula orci quis ex. Phasellus imperdiet dolor at luctus tempor. Curabitur nisi enim, porta ut gravida nec, feugiat fermentum purus. Donec hendrerit justo metus. In ultrices malesuada erat id scelerisque. Sed sapien nisi, feugiat in ligula vitae, condimentum accumsan nisi. Nunc sit amet est leo. Quisque hendrerit, libero ut viverra aliquet, neque mi vestibulum mauris, a tincidunt nulla lacus vitae nunc. Cras eros ex, tincidunt ac porta et, vulputate ut lectus. Curabitur ultricies faucibus turpis, ac placerat sem sollicitudin at. Ut libero odio, eleifend in urna non, varius imperdiet diam. Aenean lacinia dapibus mauris. Sed posuere imperdiet ipsum a fermentum.

Nulla lobortis enim ac magna rhoncus, nec condimentum erat aliquam. Nullam laoreet interdum lacus, ac rutrum eros dictum vel. Cras lobortis egestas lectus, id varius turpis rhoncus et. Nam vitae auctor ligula, et fermentum turpis. Morbi neque tellus, dignissim a cursus sed, tempus eu sapien. Morbi volutpat convallis mauris, a euismod dui egestas sit amet. Nullam a volutpat mauris. Fusce sed ipsum lectus. In feugiat, velit eu fermentum efficitur, mi ex eleifend ante, eget scelerisque sem turpis nec augue.

Vestibulum posuere nibh ut iaculis semper. Ut diam justo, interdum quis felis ac, posuere fermentum ex. Fusce tincidunt vel nunc non semper. Sed ultrices suscipit dui, vel lacinia lorem euismod quis. Etiam pellentesque vitae sem eu bibendum. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Pellentesque scelerisque congue ullamcorper. Sed vehicula sodales velit a scelerisque. Pellentesque dignissim lectus ipsum, quis consectetur tellus rhoncus a.

Nunc placerat ut lectus vel ornare. Sed nec dictum enim. Donec imperdiet, ipsum ut facilisis blandit, lacus nisi maximus ex, sed semper nisl metus eget leo. Nunc efficitur risus ac risus placerat, vel ullamcorper felis interdum. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Duis vitae felis vel nibh sodales fringilla. Donec semper eleifend sem quis ornare. Proin et leo ut dolor consectetur vehicula. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Nunc dignissim interdum orci, sit amet pretium nibh consectetur sagittis. Aenean a eros id risus aliquam placerat nec ut lectus. Curabitur at quam in nisi sodales imperdiet in at erat. Praesent euismod pulvinar imperdiet. Nam auctor mattis nisi in efficitur. Quisque non cursus ipsum, consequat vehicula justo. Fusce varius metus et nulla rutrum scelerisque. Praesent molestie elementum nulla a consequat. In at facilisis nisi, convallis molestie sapien. Cras id ullamcorper purus. Sed at lectus sit amet dolor finibus suscipit vel et purus. Sed odio ipsum, dictum vel justo sit amet, interdum dictum justo. Quisque euismod quam magna, at dignissim eros varius in. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Acknowledgments

Acknowledgments.tex

Contents

Abstract iii

Acknowledgments iv

Contents v

List of Figures vi

1 Introduction 2
- 1.1 Motivation . . . 2
- 1.2 Aim . . . 3
- 1.3 Research questions . . . 3
- 1.4 Delimitations . . . 3

2 Theory 4
- 2.1 Artificial Neural Networks . . . 4
- 2.2 Reinforcement Learning . . . 4
- 2.3 Related Work . . . 5

3 Method 8
- 3.1 Problem Statement . . . 8
- 3.2 Environment . . . 8
- 3.3 Algorithm . . . 9
- 3.4 Architecture . . . 9
- 3.5 Implementation . . . 10
- 3.6 Experiments . . . 10

4 Results 11

5 Discussion 12
- 5.1 Results . . . 12
- 5.2 Method . . . 12
- 5.3 The work in a wider context . . . 12

6 Conclusion 13

Bibliography 14

List of Figures

2.1 Partially observable Markov decision process. 5

3.1 Samples from each environment. 9

Notation

x	vector
X	matrix or set
$\mathbb{X}$	index set

1 Introduction

In this thesis project, the problem of searching for targets in unknown but familiar environments is addressed. This chapter presents the motivation behind the project, the research questions that are addressed, and the delimitations.

1.1 Motivation

The ability to visually search for targets in an environment is crucial to many parts of our daily lives. We are constantly looking for things, be it the right book in the bookshelf, a certain keyword in an article or blueberries in the forest. In many cases, it is important that this search is efficient and fast. Animals need to quickly identify predators, and drivers need to be able to search for pedestrians crossing the road they are driving on.

While searching for targets is often seemingly effortless to humans, it is a complex process. How humans and animals search for things has been extensively studied in neuroscience and neurobiology [3, 6, 5]. Applications from automated search and rescue to helping robots mean that it is of great interest to automate visual search. In the computer vision field, there has been several attempts to mimic the way humans search in machines []. Most attempts focus on fully observable scenes where the target is in view and the task is to localize it (object localization). However, in many real-world visual search scenarios the field-of-view is limited. This means that the search process is split into two steps: directing the field of view (covert attention), and locating targets within the view (overt attention). Much work has been focused on latter, locating targets within the field of view [].

When only a fraction of the environment is visible, where to move the field of view becomes an important decision. The characteristics of the searched environment can often be used to find targets quicker. For example, if one is foraging for blueberries it makes sense to search the ground rather than the trees. Similarly, if one is searching a satellite image for boats it is reasonable to focus on ocean shores. If you see a railroad track or the wake of a boat you can usually follow it to find a vehicle. The exact characteristics of the environment need not be constant - forests with blueberries can vary greatly in appearance and boats can be found in all of the seven seas. In many cases, the environment is familiar in that it has characteristics that are similar to previously seen environments. Humans are able to generalize in such cases.

Manually creating search algorithms for such tasks is problematic. The appearance and distribution of targets in an environment varies greatly, and may be subtle. The visual richness of the environment itself is another problem. How can you identify useful hints from the environment to guide covert attention? Manually engineering such a platform seems infeasible. If one could instead learn the underlying from a limited set of sample environments and generalize to unseen similar environments this problem would be circumvented.

1.2 Aim

This work tries to address these issues, focusing on strategic scans of larger environments where the field of view is small relative to the environment. This is a problem that has been less studied in the literature than visual search in smaller environments. There are other factors that become increasingly important. The field-of-view of the observer is often limited, and she has to move it efficiently to find the target.

The aim of this thesis is to implement and evaluate an autonomous agent that intelligently searches its environment for targets. The agent should learn common characteristics of environments and utilize this knowledge to search for targets in new environments more effectively. Furthermore, the agent should be able to generalize to unseen environments drawn from the same distribution as the ones it has seen previously.

A specific instance of the visual search problem is considered, where the environment is searched by a pan-tilt camera fixed in place. The camera has a limited view of the environent. Automating this task is of interest for multiple reasons. Manually controlling a camera may be costly, and the performance of a human operator may be suboptimal. Crucial to the problem is generalization.

1.3 Research questions

This thesis will address the following questions:

1. How can a learning agent that does efficient visual search in familiar environments be implemented?
2. How well does the learning agent generalize to unseen but familiar environments?
3. How does the learning agent compare to an exhaustive search of the environment, frontier-based algorithm, and a human searcher?

1.4 Delimitations

This thesis will be focused on the behavioral aspects of the presented problem. To train and test agents, a simplified environment will be used. This will test the desired characteristics of the agent as presented above, but will not simulate realistic environments. For simplicity, we assume that the environment is static. We also focus on the search process and not the detection, and therefore targets will be easy to detect once visible.

2 Theory

This chapter introduces relevant theory and related work. Section 2.3.2 gives some background on active vision. Section 2.3.3 describes the problem of visual searching for targets.

2.1 Artificial Neural Networks

An artificial neural network (ANN) is a type of universal function approximator.

2.1.1 Multi-layer Perceptron

2.1.2 Long Short-Term Memory

2.1.3 Convolutional Neural Network

2.2 Reinforcement Learning

Reinforcement learning (RL) is a subfield of machine learning concerned with learning from interaction how to achieve a goal. An *agent* and its *environment* interact continually over discrete time steps. At each time step the agent selects some *action* that updates the state of the environment, and gives it a *reward*. The agent selects actions using a stochastic *policy* with the goal of maximizing the *return* which is usually defined as the discounted sum of future rewards. [4]

2.2.1 Markov Decision Process

The RL setup is usually formalized as a (finite) Markov decision process (MDP).

The problem of learning from interaction to achieve a goal is usually framed as a (finite) Markov Decision Process (MDP). For regular MDPs it is assumed that the learning agent has access to some representation of the underlying *state* of the environment which it uses to select *actions*. For many problems this is not true. A partially observable Markov decision process (POMDP) is a generalization of an MDP in which it is assumed that the environment has some well defined underlying latent state, but the agent only perceives a partial *observation* of it from the environment.

A POMDP is formally defined as a 7-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{R}, \mathcal{T}, \Omega, \gamma \rangle$, where

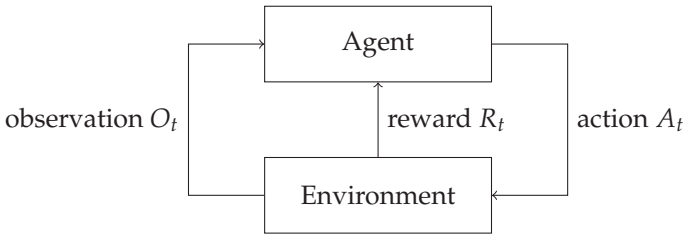

Figure 2.1: Partially observable Markov decision process.

- $\mathcal{S}$ is a finite set of states,
- $\mathcal{A}$ is a finite set of actions,
- $\mathcal{T}$ is a set of conditional state transition probabilities,
- $\mathcal{R} : S \times A \to \mathbb{R}$ is a reward function,
- Ω is a finite set of observations,
- $\mathcal{O}$ is a set of conditional observation probabilities, and
- $\gamma \in [0, 1]$ is a discount factor.

The agent interacts with the environment at discrete time steps $t = 0, 1, 2, \ldots T$. At each time step t, the agent receives an observation of the environment's state $O_t \in \Omega$ and selects some action $A_t \in \mathcal{A}$. In the next time step the agent receives a reward

action $a \in \mathcal{A}$ which causes the environment to transition to state s' with probability $\mathcal{T}(s'|s, a)$. It receives an observation $o \in \Omega$ with probability $\mathcal{O}(o|s', a)$, as well as a reward r given by $\mathcal{R}(s, a)$.

This interaction is repeated until the end of the episode at time step T. The goal of the agent is to maximize the *discounted return*, defined as the discounted sum of future rewards $G_t \doteq \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ where γ reflects the uncertainty of the environment.

Planning in a POMDP is undecidable, and solving them is often computationally intractable. Approximate solutions are more common.

2.2.2 Policies and Value Functions

Most RL algorithms estimate both a *value function* that tells the agent how good it is to be in a given state, and a

2.3 Related Work

2.3.1 Local Search

2.3.2 Active Vision

Much of past and present research in machine perception involves a passive observer. Images are passively sampled and perceived. Animal perception, however, is active. We do not only see things, but look for them. One might ask why this is the case, if there is any advantage that an active observer has over a passive one. Aloimonos and Weiss (1988) [1] introduce the paradigm called *active vision*, and prove that an active observer can solve several basic vision problems in a more efficient way than a passive one.

Bajcsy (1988) **[bajcsy_1988]** defines active vision, and perception in general, as a problem of intelligent data acquisition. An active observer needs to define and measure parameters

and errors from its scene and feed then back to control the data acquisition process. Bajscy states that one of the difficulties of this problem is that they are scene and context dependent. A thorough understanding of the data acquisition parameters and the goal of the visual processing is needed. One view lacks information that may be present with multiple views. Multiple views also add the time dimension into the problem.

In a re-visitation of active perception, Bajcsy, Aloimonos and Tsotsos (2018) [**bajcsy_aloimonos_tsotsos_2018**] stress that despite recent successes in robotics, artificial intelligence and computer vision, an intelligent agent must include active perception:

> An agent is an active perceiver if it knows why it wishes to sense, and then chooses what to perceive, and determines how, when and where to achieve that perception
>
> [**bajcsy_aloimonos_tsotsos_2018**]

2.3.3 Visual Search

The perceptual task of searching for something in a visual environment is usually referred to as *visual search*. The searched object or feature is the *target*, and the other objects or features in the environment are the *distractors*. This task has been studied extensively in psychology and neuroscience.

Wolfe (2021) [5] describes a model of visual search

Eckstein (2011) [3] reviews efforts from various subfields and identifies a set of mechanisms used to achieve efficient visual search. Knowledge about the target, distractor, background statistical properties, location probabilities, contextual cues, rewards and target prevalence are all identified as useful. This is motivated with evidence from psychology as well as neural correlates.

Visual search is not always instant, and can in fact often be slow. This is in part due to processing: our visual system cannot process the entire visual field and

Wolfe and Horowitz (2017) [**wolfe_horowitz_2017**] identify and measure a set of factors that guide attention in visual search. One of these is bottom-up guidance, in which some visual properties of the scene draw more attention than others. Another is top-down guidance, which is user driven and directed to objects with known features of desired targets. Scene guidance is also identified, in which attributes of the scene guide attention to areas likely to contain targets.

These works ground the task considered in this project in psychology.

2.3.4 Object Detection

A similar problem can be found in the computer vision literature under *object detection*. The goal of object detection is to, given an input image, detect instances of semantic objects in it. This includes assigning a bounding box to the objects, and classifying the object. The input image is usually passively sampled, and the whole scene is visible at once.

Caicedo and Lazebnik (2015) [2] propose to use deep reinforcement learning for active object localization in images where the object to be localized is fully visible. An agent is trained to successively improve a bounding box using translating and scaling transformations. They use a reward signal that is proportional to how well the current box covers the target object. An action that improves the region is rewarded with +1, and -1 otherwise. Without this quantization, the difference was small enough to confuse the agent. Binary rewards communicate more clearly which transformations keep the object inside the box and which take the box away from the target. When there is no action that improves the bounding box, the agent may select a trigger action (which would be the only action that does not give a negative reward) which resets the box. This way the agent may select additional bounding boxes. Each trigger modifies the environment by marking it so that the agent may learn to not select the

same region twice. This is referred to as an inhibition-of-return mechanism, and is widely used in visual attention models **[[16] in caicedo_active_2015]**. This method has a few shortcomings for the problem considered in this project. The object may not be visible in the initial frame so the agent cannot act in the same way.

A separate field is active object search, which is perhaps most closely related to the problem we consider in this work. In active object search,

A similar work by Ghesu et al. (2016) [**ghesu_artificial_2016**] present an agent for anatomical landmark detection trained with DRL. Different from [2] is that the entire scene is not visible at once. The agent sees a limited region of interest in an image, with its center representing the current position of the agent. The actions available to the agent translate the view up, down, left and aright. A reward is given to the agent that is equal to the supervised relative distance-change to the landmark after each action. Three datasets of 891 anatomical images are used. The agent starts at random positions in the image close to the target landmark and is tasked with moving to the target location. While achieving strong results (90% success rate), the scenes and targets are all drawn from a distribution with low variance. Most real-world search tasks exhibit larger variance than anatomical images of the human body.

Zhu et al. (2016) [8] create a model for target-driven visual navigation in indooor scenes with DRL. An observer is given a partial image of its scene as well as an image of the target object, and is tasked with navigating to the object n the scene with a minimal number of steps. The agent moves forwards, backwards, and turns left and right at constant step lengths. They use a reward signal with a small time penalty to incentivize task completion in few steps. They compare their approach to random walk and the shortest path and achieve promising results. This setup is quite similar to the one considered in this report, but the authors make a few assumptions that we do not. They a set of 32 scenes, each of which contain a fixed number of object instances. They focus on learning spatial relationships between objects in these specific scenes, and have scene-specific layers to achieve this. Thus, while they show that they can adapt a trained network to a new scene, their approach is unable to zero-shot generalize to new scenes.

A similar work by Ye et al. (2018) [7] integrates an object recognition module with a deep reinforcement learning based visual navigation module. They experiment with a set of reward functions and find that constant time penalizing rewards can be problematic and lead to slow convergence. Their experiments make the same assumptions as [**zhu_target_driven**] - the scenes and targets used during testing have all been seen during training.

2.3.5 Visual Attention

2.3.5.1 Exploration and Exploitation Trade-off

2.3.5.2 Generalization in Deep Reinforcement Learning

Kobbe et al. (2020) [] study generalization in RL. They introduce a benchmark of procedurally generated i.i.d. environments, and find that this is essential to

3 Method

In this chapter, the method used is described. Section 3.1 formalizes the problem solved. Section 3.2 details the environment used to train and test an agent. Section 3.3 describes the algorithm used to train the agent.

3.1 Problem Statement

The problem of finding an optimal sequence of actions to find can be cast as a partially observable Markov decision process (POMDP).

Formally, we let the environment's scene be described by an n-dimensional discrete Euclidean space. In the environment, there are N targets, each described by a subspace. The agent observes a subspace of the state and can, though a discrete set of actions, transform this subspace. This corresponds to changing the field of perception. With a final action, the agent can indicate that it thinks that a target space intersects with the observed subspace. The goal of the agent is to minimize the number of actions until all N targets have been found.

We make the assumption that the agent has knowledge of the shape of its environment's space and its current position at each time step. Using this information it can determine where it has been previously.

3.2 Environment

In this project, we focus on search in the visual domain. We let the state space be a three-dimensional of shape (H, W, C) where H is the height of the space, W is the width of the space, and C is the number of channels in the space. The observation space is three-dimensional of shape (H_v, H_w, C). We let the action space consist of four view-transforming actions that translate the observed subspace up, down, left or right along. This roughly corresponds to panning and tilting a camera to look around in an environment. It is a discretization and simplification of such a task in a 3D environment

The environments to be searched are drawn from a distribution, with varying but similar appearance, target locations and appearances. For all environments, the appearance correlates to the probability of targets.

To incentivize finding targets quickly, the reward signal is set to -1 for each time step. Since viewing a window twice is redundant, such actions are punished by setting the reward to -2.

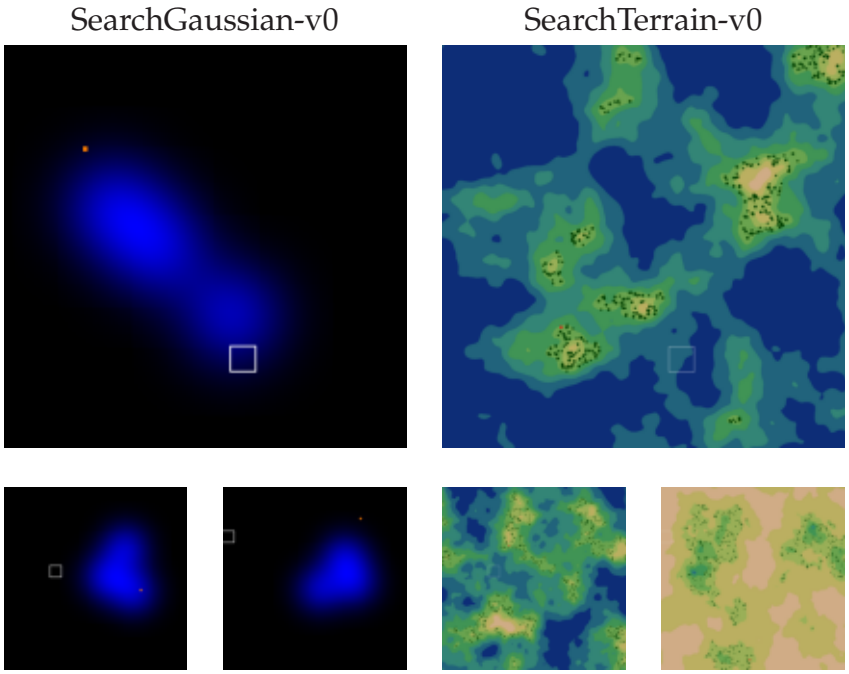

Figure 3.1: Samples from each environment.

If the agent selects the trigger action when a target overlaps with the window, the reward is set to 5. When all targets have been triggered, or when 1000 time steps have passed, the episode ends.

To test the capability of the method, we use three environments of varying difficulty. The first environment visually simple ... The second environment is ... The third environment consists of real images for object localization. The images are from the dataset [cite] and consists of N training samples and M test samples. Each image is 2440x2440 pixels and is therefore expensive to run object detection algorithms on. There are N training samples and M test samples.

Note that all environments are static, in that the scene is unaffected by the actions of the agent. The methods presented do, however, apply to other unknown dynamics.

3.3 Algorithm

The agent is trained with reinforcement learning using Proximal Policy Optimization with function approximation using neural networks.

3.4 Architecture

The architecture of the neural network is presented in Figure X.

The network is split into three main section: the feature extraction, the shared network, and the policy and value heads. Like Mnih et al., we use a CNN [] for feature extraction. The shared network consists of an LSTM **[hochreiter]**. Finally, both the policy and the value head are two-layer MLPs.

3.5 Implementation

The environment is implemented with Gym [**gym**], and the agent is implemented with PyTorch [**pytorch**].

3.6 Experiments

The first environment was used to determine a good observation space. By just observing the current window, the agent can never learn a suitable policy to solve the problem. This is because it cannot distinguish between equal windows at different locations. Experiments are run with several additional observation types: window position, ... Results of these experiments are presented for this environment only.

The agent was trained using the algorithm described in Section 3.3 for 100 million time steps in all three environments using PPO, PPG and A2C. Hyperparameters are tuned with random search separately for each environment. For all experiments, the average return per episode is reported together with the theoretically optimal reward (obtained with an optimal path). The agent was trained and tested on the full distribution of environments.

With the best performing algorithm we compare three different reward signals. One gives a constant time penalty of -1 to incentivize finding the target quickly. The second gives a constant time penalty of -2 and an exploration bonus of +1 when a new view is reached. The third gives a reward of +1 for moving closer to the target and a penalty of -1 otherwise, as in [2].

Additionally, experiments to eveluate the generalization capability of the agent were conducted. These were conducted on the procedurally generated terrain environment following the approach suggested in [**procgen**]. During training, the seed pool size was fixed to various sizes to limit the training set size. The agent was trained for varying number of timesteps and then tested on the full distribution of environments. This way, we can get a sense of how much data and simulation is required to use the approach for real-world tasks.

All experiments are conducted on an Intel Core i9-10900X CPU and an NVIDIA GeForce RTX 2080 Ti GPU.

For each experiment, we report the mean return and episode length over time during training. We compare the approach to an exhaustive search and a human searcher with prior knowledge of the characteristics of the searched environments. As per [rlliable], we report results across multiple seeds.

4 Results

When it comes to hyperparameters, we find that letting the number of rollout steps be substantially lower than the episode length we achieve much more stable training results. Furthermore, increasing the number of weights in the neural network made it more difficult to train.

We find that the hyperparameters from [procgen] perform well, especially when the number of environments is large.

This coupled with a sparse reward signal led to many cases where the agent converged towards a poor local optimum (or perhaps never converged at all).

5 Discussion

This chapter contains the following sub-headings.

5.1 Results

5.2 Method

It is worth considering whether using a learning agent like this is suitable for this task. One could imagine that it is possible to compute an optimal strategy for certain environments. However, this quickly falls apart. The dynamics of environments can vary considerably which may drastically affect how a manual approach is implemented.

Another thing worth discussing is the possibility of combining manual search method with reinforcement learning. One could imagine combining a frontier based approach with a learning approach.

In this work, we have only covered searches where the view is transformed in the spatial domain. However, the method could be applied to a broader category of problems. For instance, one could imagine a scenario when searching along the time dimension is useful. If we let the actions be translations arbitrary translations along the time dimensions in, say, a long audio or video file, the agent could learn to look for landmark features in such modalities.

5.3 The work in a wider context

While automated search systems have many positive uses, like XXX, there are certainly other use cases that could be considered negative. Mass surveillance, XXX, are both very relevant today.

6 Conclusion

Bibliography

[1] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. "Active vision". In: *International Journal of Computer Vision* 1.4 (Jan. 1988). 705 citations (Crossref) [2022-02-07], pp. 333–356. ISSN: 0920-5691, 1573-1405. DOI: 10/cn4mdc. URL: http://link.springer.com/10.1007/BF00133571 (visited on 02/07/2022).

[2] Juan C. Caicedo and Svetlana Lazebnik. "Active Object Localization with Deep Reinforcement Learning". In: *arXiv:1511.06015 [cs]* (Nov. 18, 2015). arXiv: 1511.06015. URL: http://arxiv.org/abs/1511.06015 (visited on 02/03/2022).

[3] M. P. Eckstein. "Visual search: A retrospective". In: *Journal of Vision* 11.5 (Dec. 30, 2011). 207 citations (Crossref) [2022-02-28], pp. 14–14. ISSN: 1534-7362. DOI: 10.1167/11.5.14. URL: http://jov.arvojournals.org/Article.aspx?doi=10.1167/11.5.14 (visited on 02/22/2022).

[4] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction*. Second edition. Adaptive computation and machine learning series. Cambridge, Massachusetts: The MIT Press, 2018. 526 pp. ISBN: 978-0-262-03924-6.

[5] Jeremy M. Wolfe. "Guided Search 6.0: An updated model of visual search". In: *Psychonomic Bulletin & Review* 28.4 (Aug. 2021). 29 citations (Crossref) [2022-03-02], pp. 1060–1092. ISSN: 1531-5320. DOI: 10.3758/s13423-020-01859-9.

[6] Jeremy M. Wolfe. "Visual search". In: *Current biology : CB* 20.8 (Apr. 27, 2010). 64 citations (Crossref) [2022-03-02] Publisher: NIH Public Access, R346. DOI: 10.1016/j.cub.2010.02.016. URL: https://www.ncbi.nlm.nih.gov/labs/pmc/articles/PMC5678963/ (visited on 03/02/2022).

[7] Xin Ye, Zhe Lin, Haoxiang Li, Shibin Zheng, and Yezhou Yang. "Active Object Perceiver: Recognition-Guided Policy Learning for Object Searching on Mobile Robots". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). ISSN: 2153-0866. Oct. 2018, pp. 6857–6863. DOI: 10.1109/IROS.2018.8593720.

[8] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J. Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. "Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning". In: *arXiv:1609.05143 [cs]* (Sept. 16, 2016). arXiv: 1609.05143. URL: http://arxiv.org/abs/1609.05143 (visited on 03/14/2022).