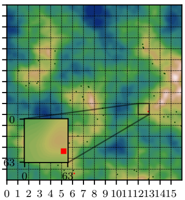
Learning to Search for Targets

with Deep Reinforcement Learning

Oskar Lundin

Linköping University

April 20, 2022



Outline

Problem Description

Autonomous search for a set of targets in an environment with a fixed camera.

- Agent observes a limited region of the environment.
- Can direct its gaze and indicate when a target is in view through actions.
- Should locate all targets while minimizing the number of actions.
- Applications in search and rescue, fire detection, surveillance, etc.

- In a random environment with uniformly distributed targets, random or exhaustive search is sufficient.
- Most real-world search tasks are not random, but exhibit structure.
- Cues in the searched scene can be used to find targets quicker.
  - Books are in bookshelves.
  - Cars can be found on roads.
  - Some targets spread out
  - Some are close together.
- Patterns and cues may be subtle and difficult to pick up.
- Manually engineering a searching system with rules can be difficult and costly.
- Can a system learn to search intelligently from a set of samples and generalize to similar search tasks?

Challenges

- Prioritize regions with high probability of targets based on previous experience.
- Find multiple targets while minimizing path length.
- Search exhaustively while avoiding searching the same region twice.

Research Questions

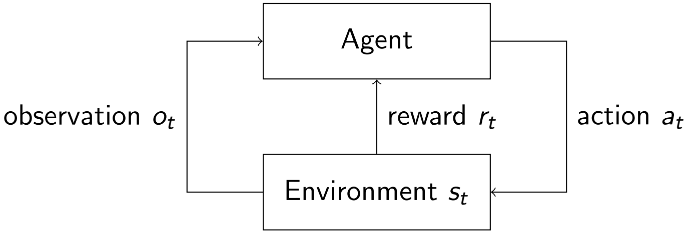
1. How can an agent that learns to intelligently search for targets be implemented with reinforcement learning?
2. How does the learning agent compare to random walk, exhaustive search and a human searcher with prior knowledge of the searched scene?
3. How does the agent's ability to generalize to unseen in-distribution environments depend on the number of training samples?

Reinforcement Learning

Reinforcement learning (RL) is a paradigm for learning mappings from observations to actions.

Partially observable Markov decision process (POMDP):

- Agent interacts with environment over discrete time steps $t = 0, 1, 2 \ldots$
- Takes action a_t in state s_t
- Perceives observation o_t
- New state s_{t+1}



Search with Reinforcement Learning

. . .

Problem Formulation

- Agent searches scene $S \subset \mathbb{R}^d$.
- Scene contains set of targets $\{t_0, \ldots t_n\}$, $t_i \in S$.
- Agent perceives view $V \subset S$.
- Move actions transform view to new subspace.
- Trigger action indicates that target(s) is in view.
- **Locate all targets while minimizing the number of time steps**.

Environments

- Three environments with varying characteristics.
- Search space discretized into 16×16 camera positions.
- Each camera position has a unique view $V \subset S$.
- Three targets in all scenes.
- Target probability correlated with scene appearance.
- Should be possible to do better than exhaustive search on average.
- Scenes procedurally generated:
    - Pseudorandom seed determines scene appearance and target positions.
    - Gives control over difficulty to solve.
    - Can vary training and test set sizes.

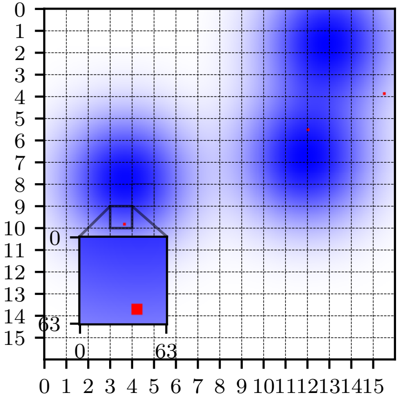
Observation, Action and Reward

At each time step t:

- The agent receives observation $o_t = \langle x_t, p_t \rangle$, where
  - $x_t \in \mathbb{R}^{3\times 64\times 64}$ is an RGB image of current view, and
  - $p_t \in \{0, \ldots, 15\} \times \{0, \ldots, 15\}$ is the position of the camera.
- Takes action $a_t \in \{\mathtt{TRIGGER}, \mathtt{UP}, \mathtt{DOWN}, \mathtt{LEFT}, \mathtt{RIGHT}\}$, where
  - `TRIGGER` indicates that a target is in view, and
  - `UP`, `DOWN`, `LEFT`, `RIGHT` move the view in each cardinal direction.
- Receives reward $r_t = 10h - 1$ where $h = |T \cap V|$ is the number of targets in view.
  - Rewarded for finding targets.
  - Constant penalty encourages quick episode completion.
  - Two variants that reward exploration or moving towards targets: r'_t and r''_t

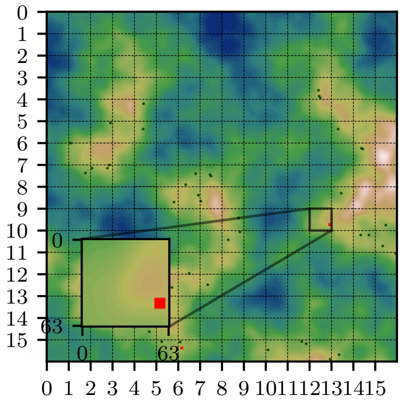
Gaussian Environment

- Two-dimensional scene.
- Three gaussian kernels with random center.
- Sum of kernels determine appearance of scene and probability of targets.
- Clear correlation between appearance and desired behavior.



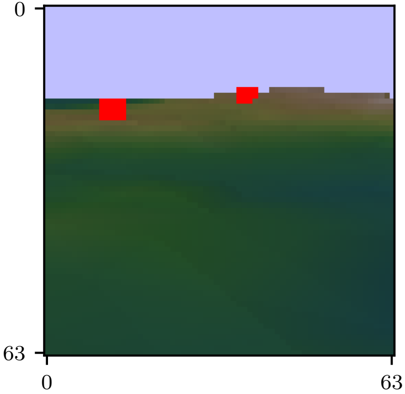
Terrain Environment

- Similar to previous environment.
- Terrain seen from above.
- Gradient noise used to generate height map.
- Color determined by height.
- Targets placed with uniform probability across coastlines.
- More realistic, higher variance.
- Analogous to search and rescue with UAV.



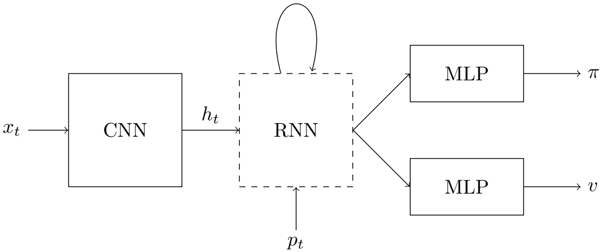
Camera Environment

- 3D scene viewed from a perspective projection camera.
- Height map from terrain environment turned into mesh, same appearance and target probability as before.
- Camera location fixed at center of scene.
- Moving actions control pan and tilt (pitch and yaw).
- Visually complex, difficult to interpret.



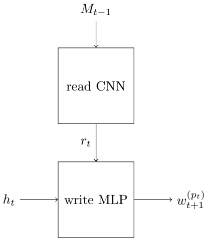
Architecture

- Actor-critic method.
- Trained with proximal policy optimization.
- Image x_t passed through CNN.
- Latent image representation h_t and position p_t passed through RNN.
  1. LSTM with input $[h_t, p_t]$.
  2. **Spatial memory.**
- Policy head approximates π with MLP.
- Value head approximates v with MLP.



Spatial Memory

- LSTM may have difficulties remembering over many time steps and reasoning over spatial relations.
- Specialized memory could be more useful.
- Structured memory with one slot for each camera position.
- Memory read with CNN.
- Written to with previous read and new latent image.



Experiments

- Train for 25M time steps.
- Results reported across 3 seeds.
- Separate training and test sets.
- Same hyperparameters in all runs.

Search space:

- Larger search spaces places stronger demands on memory
- Scene understanding, remember searched positions.
- Investigate impact by comparing agents on 8×8, 12×12, 16×16 versions of gaussian environment.

Experiments

Generalization:

- Limit number of scene samples seen during training to 100, 1000, 10 000.
- Use terrain environment, high appearance variance and somewhat realistic.
- Fix seed pool used to generate scenes seen during training.
- Train agents until convergence (or for a fixed number of time steps).
- Test on held out scenes from full distribution.

Implementation

- OpenAI Gym environment interface.
- PyTorch for models and automatic differentiation.

Preliminary Results

Problems:

- Many configurable parts (hyperparameters, environment settings, agent architectures).
- Long training times, difficult to predict hyperparameter interplay, expensive to tune.
- Difficult to design reward signal, magnitudes matter.
- Collecting rigorous results across multiple seeds will take time.
- Spatial memory approach can handle it.

Future Steps

1. Collect complete results across multiple seeds for all experiments.
2. More baselines vs. ablation studies?
3. Tidy up report.
4. Discussion and conclusion.
5. Presentation preparation.