UAV Path Planning for Wireless Data Harvesting: A Deep Reinforcement Learning Approach

Harald Bayerlein[1], Mirco Theile[2], Marco Caccamo[2], and David Gesbert[1]

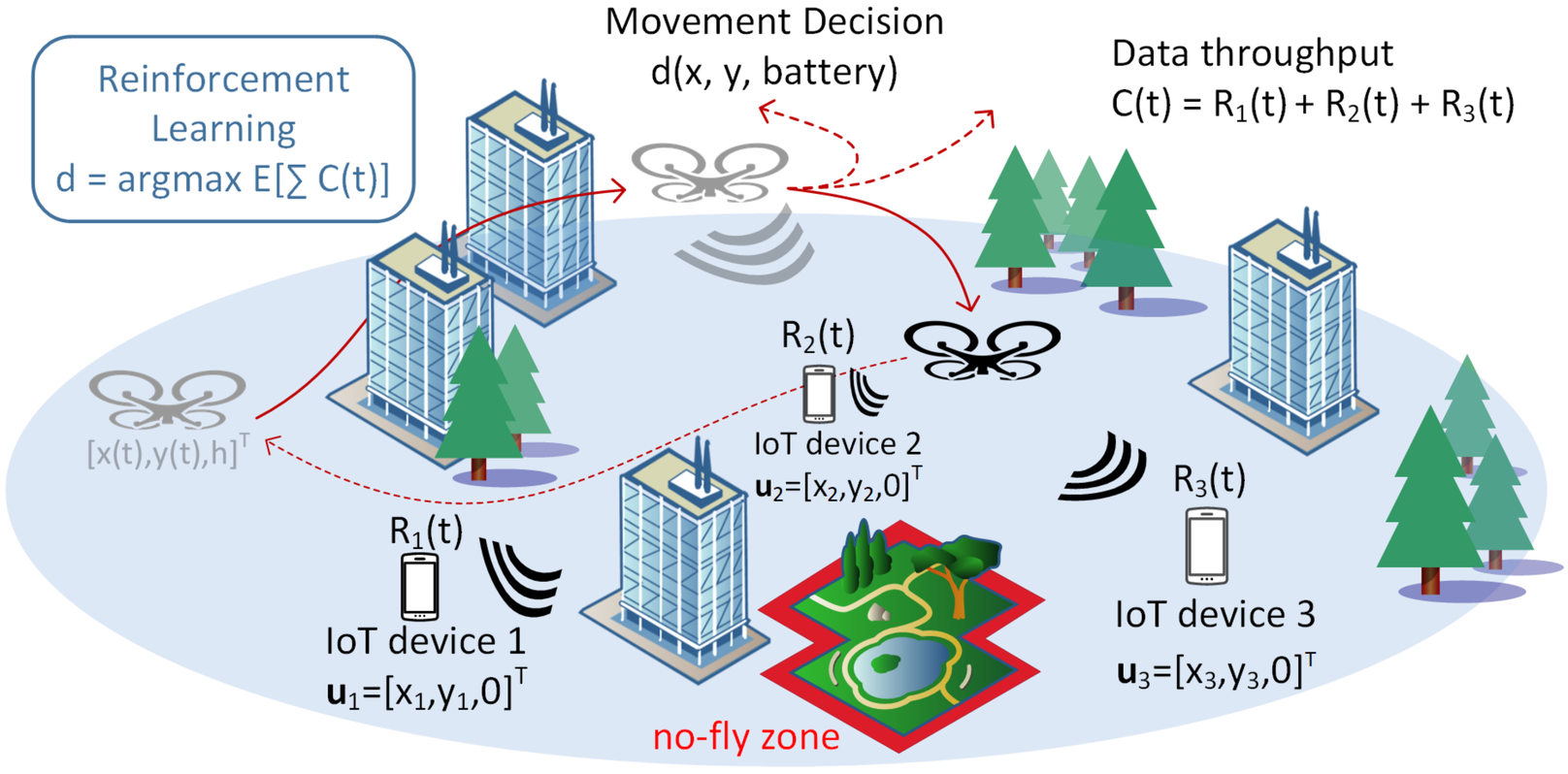
[1]Communication Systems Department, EURECOM, France

[2]TUM Department of Mechanical Engineering, Technical University of Munich, Germany

Autonomous UAV Data Harvesting [1]

- Quadcopter UAV carrying a wireless access point collects data from Internet of Things (IoT) devices
- Trajectory planning subject to constraints on flying time, obstacle avoidance and regulatory no-fly zones (NFZs)
- Changes in scenario (position of IoT devices, amount of data to be picked up etc.) usually require expensive recomputation or relearning

➥ Use reinforcement learning (RL) to learn a UAV control policy that generalizes over changing scenario parameters.



System Model

- Square grid world $M \times M \in \mathbb{N}^2$ with K static IoT devices at positions $\mathbf{u}_k = [x_k, y_k, 0]^\mathrm{T} \in \mathbb{R}^3$
- Maximum flying time $T \in \mathbb{N}$ discretized into mission time slots $t \in [0, T]$
- UAV's position $[x(t), y(t), h]^\mathrm{T} \in \mathbb{R}^3$ and velocity $v(t) \in \{0, V\}$
- Information rate for k-th device:

$$R_k(t) = \log_2\left(1 + \mathrm{SNR}_k(t)\right)$$

$$\mathrm{SNR}_k(t) = \frac{P_k}{\sigma^2} \cdot d_k(t)^{-\alpha_l} \cdot 10^{\eta_l/10}$$

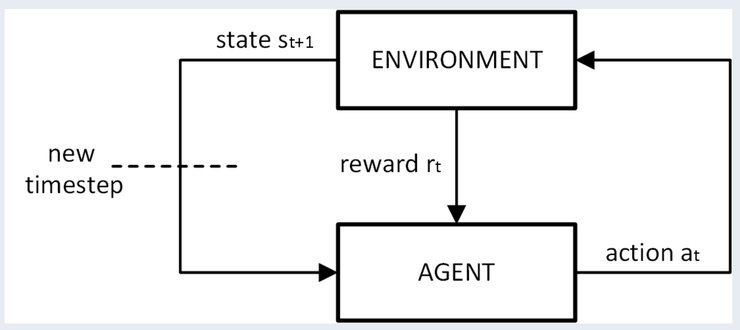
- Scheduling algorithm based on TDMA

➥ Throughput maximization over K devices:

$$\max_{x(t),y(t)} \sum_{t=0}^{T} C(t) = \max_{x(t),y(t)} \sum_{t=0}^{T} \sum_{k=1}^{K} R_k(t)$$

Reinforcement Learning [2]

Main idea: an *agent* in an environment takes *actions* trying to maximize *reward*



- Modelled as finite MDP $(\mathcal{S}, \mathcal{A}, R, P)$
- Probabilistic policy $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$
- *Q-function* or state action-value function:

$$Q^\pi(s, a) = \mathbb{E}_\pi\left[R_t | s_t = s, a_t = a\right]$$

➥ Optimal policy $\pi^*(a|s) = \mathrm{argmax}_a\, Q^{\pi^*}(s, a)$

Double Deep Q-Learning [3]

A *double deep Q-network (DDQN)* parameterizing the Q-function with parameter vector θ is trained to minimize the expected temporal difference (TD) error given by

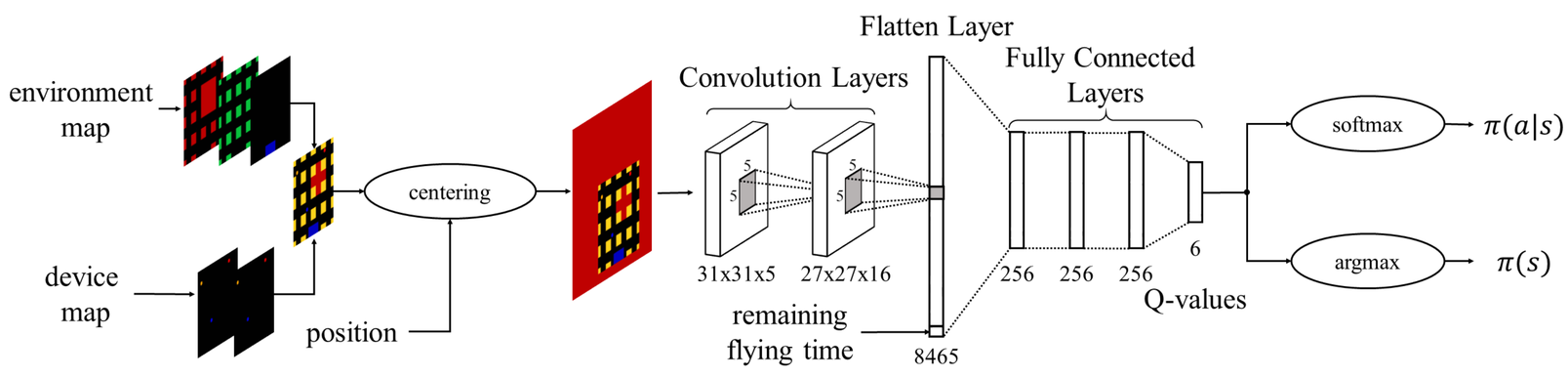
$$L(\theta) = \mathbb{E}_{s,a,s' \sim \mathcal{D}}[(Q_\theta(s, a) - Y(s, a, s'))^2]$$

where the target value, computed using a separate target network with parameters $\bar{\theta}$, is given by

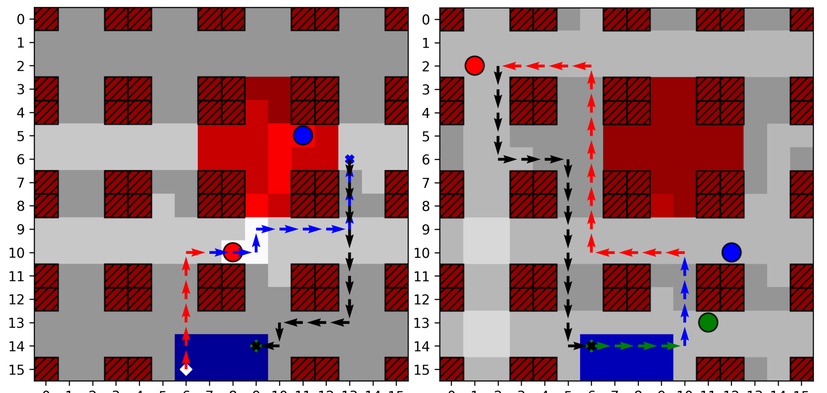
$$Y(s, a, s') = r(s, a) + \gamma Q_{\bar{\theta}}(s', \mathrm{argmax}_{a'}\, Q_\theta(s', a')).$$

We added some other extensions to the training process, i.a. *combined experience replay* [4], as a remedy for the agent's sensitivity to the selection of the replay buffer size.

DQN Architecture with Map Centering



"Manhattan"-type Map



(a) Time 31/38; Data 17.4/17.4

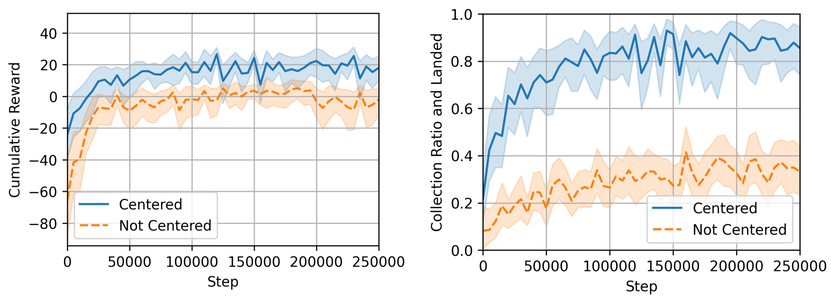
(b) Time 41/41; Data 34.5/34.5

(c) Time 30/35; Data 32.0/50.6

(d) Time 45/65; Data 60.7/60.7

Figure: Illustration of the same agent adapting to differences in device count and device placement as well as flight time limits, showing used and available flying time and collected and available total data in the Manhattan scenario.

Advantage of Map Centering



(a) Episodic cumulative reward (b) Collection ratio and landed

Figure: Training process comparison between centered and non-centered map input showing the average and 99% quantiles of three training processes each, with episodic metrics grouped in bins of 5000 step width.

Metric	Manhattan Map
Has Landed	99.5%
Collection Ratio	94.8%
Collection Ratio and Landed	94.6%

Table: Performance metrics averaged over 1000 random scenario Monte Carlo iterations.

References

[1] H. Bayerlein, M. Theile, M. Caccamo, and D. Gesbert "UAV Path Planning for Wireless Data Harvesting: A Deep Reinforcement Learning Approach," arXiv:2007.00544 [cs.LG].

[2] R. S. Sutton and A. G. Barto, Reinforcement Learning: an introduction. MIT Press, second ed., 2018.

[3] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in Thirtieth AAAI conference on artificial intelligence, pp. 2094–2100, 2016.

[4] S. Zhang and R. S. Sutton, "A deeper look at experience replay," arXiv:1712.01275 [cs.LG], 2017.