IEEE GLOBAL COMMUNICATIONS CONFERENCE
(IEEE GLOBECOM 2023)

Intelligent Execution of Computer Vision Tasks in Delay-Constrained UAV-aided Networks

Paper ID: 1570939012

Nancy Varshney[*], Corrado Puligheddu [†], Carla Fabiana Chiasserini[†], Claudio Casetti[†], and Swades De[*]

[*] Department of Electrical Engineering, Indian Institute of Technology Delhi, India
[†]Department of Electronics and Telecommunications, Politecnico di Torino, Italy

Outline

Introduction

- UAVs find dominant application in surveillance
- Context-awareness allows UAVs to operate effectively and reliably in complex environments
- Cavaliere et al.[1] presented a UAV-based surveillance system with a UAV having cognitive abilities for situational awareness
- However, UAV operation requires energy-efficient and low-latency object/video detection
- Local processing at UAV requires balance between system objectives and limited resources, e.g., power, bandwidth, computation
- UAVs need to communicate data, such as sensor readings, status updates, and control commands to ground control stations (GCS) or edge server that in turns control UAV motion and operation

[1] D. Cavaliere *et al.*, "Proactive uavs for cognitive contextual awareness", *IEEE Systems Journal*, vol. 13, no. 3, pp. 3568–3579, 2018.

Motivation

- State-of-art techniques on video object detection use deep neural networks (DNNs) that are computation, memory, and power intensive[2]
- Energy consumption for UAV's communication is a major challenge, as it directly affects its airborne operation
- Critical frames having object of interest form a small fraction of total video, but their timely detection (equivalent to low packet drop rate) is crucial for delay-constrained autonomous UAV operations
- mmWave technology offers high bandwidth for transmission but highly it is susceptible to blockages
- Advantageous to process delay constrained critical frames locally at the UAV in poor communication channel conditions
- Execution of computer vision tasks based on frame semantics necessitate running object detection algorithm, which is energy intensive
- Offloading local computation at the UAV requires high communication overhead
- Balancing frame drop rate with UAV power consumption is crucial
- Autonomous UAV operation is limited by the computational/energy ability of UAV and the low latency requirements of the data processing and transmission
    - Deciding on processing or offloading data under delay and power constraints in fading channel conditions is challenging

[2] J. Wang *et al.*, "Bandwidth-efficient live video analytics for drones via edge computing", in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, IEEE, 2018, pp. 159–173.

Related Works

- For energy-efficient UAV surveillance majority of the existing works in literature targeted optimizing trajectory, resource allocation, and using UAV swarm[3]
- Huda et al.[4] surveyed UAV computation offloading decisions for various scenarios involving service latency, energy/power consumption, and execution delay
- Motlagh et al.[5] studied off-loading of a video data processing to an edge node compared to local processing in UAV with a focus on processing delay and energy consumption
- Bai et al.[6] proposed energy efficient offloading solution that can perform under potential eavesdroppers with a focus on UAV energy consumption and computational load
- Hayat et al.[7] studied different modes for image processing based on whether and what to offload, based on the network conditions and processing time only

[3] F. Koohifar *et al.*, "Receding horizon multi-UAV cooperative tracking of moving RF source", *IEEE Communications Letters*, vol. 21, no. 6, pp. 1433–1436, 2016.

[4] S. A. Huda and S. Moh, "Survey on computation offloading in UAV-Enabled mobile edge computing", *Journal of Network and Computer Applications*, vol. 201, p. 103 341, 2022.

[5] N. H. Motlagh *et al.*, "UAV-based IoT platform: A crowd surveillance use case", *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, 2017.

[6] T. Bai *et al.*, "Energy-efficient computation offloading for secure UAV-edge-computing systems", *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 6074–6087, 2019.

[7] S. Hayat *et al.*, "Edge computing in 5G for drone navigation: What to offload?", *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2571–2578, 2021.

Research Gaps and Contributions

- Exiting literature did not investigate
  - UAV offloading strategies considering communication channel constraints
  - criticality of data content when making decisions regarding data offloading or local processing
- Need to design a UAV-aided monitoring mechanism while taking into account the following factors
  - energy limitations of the UAV
  - time criticality for object recognition to allow for context-sensitive UAV operations
  - communication with channel constraints
  - data criticality states – Modeled as an MDP (Markov Decision Process)
  - frame drop rates

Contributions

- We propose a delayed-reward State-Action-Reward-State-Action (SARSA) algorithm based RL framework for optimizing offloading decisions in surveillance by a camera-equipped UAV at mmWaves
- Using statistics of channel conditions as well as statistics of frame criticality, we aim to reduce UAV energy consumption for time-constrained frames while maintaining low frame drop rates

System model

- Consider a UAV following a route $\mathcal{R}$ with a time-varying communication link to a GCS for information exchange
- UAV is equipped with a camera that captures frames at every $\Delta t = \frac{1}{\text{FPS}}$ seconds
- On the route $\mathcal{R}$ there may be critical objects or non-critical objects $\longleftarrow$ unknown to UAV
- A frame at time t can be in two states: $\{$Critical, Non-critical$\}$
- Time is divided into slots, with a slot duration equal to channel coherence time T_{Coh} (e.g., 10 ms)
- In each slot, the communication link to the GCS can be either good (i.e., perfect data transmission) or bad (i.e., complete link blockage)
- Transmission of a frame data packet can occur over one or more time slots
- Frame's complete execution occurs when the channel remains good throughout its transmission time

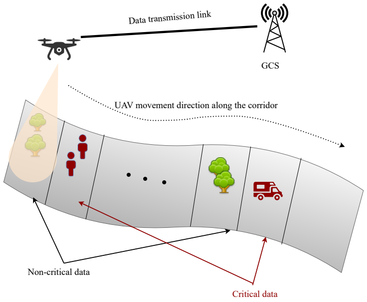


Figure 1: System model

2-state MDPs

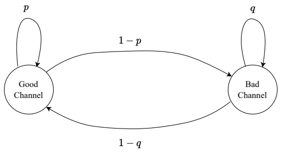

$$\begin{tikzcd} \text{Good Channel} \arrow[loop above, "p"] \arrow[r, bend left, "1-p"] & \text{Bad Channel} \arrow[loop above, "q"] \arrow[l, bend left, "1-q"] \end{tikzcd}$$

Figure 2: Two-state Markov model of channel

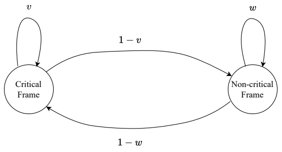

$$\begin{tikzcd} \text{Critical Frame} \arrow[loop above, "v"] \arrow[r, bend left, "1-v"] & \text{Non-critical Frame} \arrow[loop above, "w"] \arrow[l, bend left, "1-w"] \end{tikzcd}$$

Figure 3: Two-state Markov model of object criticality

- Channel transition matrix

$$P_{Ch} = \begin{bmatrix} p & 1-p \\ 1-q & q \end{bmatrix} \tag{1}$$

- Frame transition probability

$$P_F = \begin{bmatrix} v & 1-v \\ 1-w & w \end{bmatrix} \tag{2}$$

- Values of p, q and v, w determined at each slot respectively based on feedback obtained from historical channel and frame data

- When n-th frame is processed locally at the UAV $\rightarrow$ knowledge of frame's criticality status obtained after its processing time $t^{\text{UAV}}_{\text{process}}$ $\rightarrow$ delayed reward
- $n = \{1, \ldots, N_F\}$, N_F being the total number of frames over an episode of UAV surveillance
- When n-th frame is transmitted to GCS, the UAV anticipates receiving feedback within a duration of T_{wait}

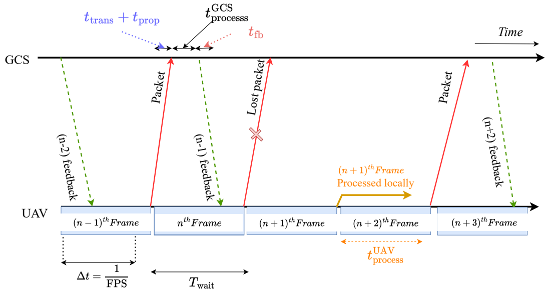


Figure 4: Flow diagram

- Total computer vision task execution time

$$t_{\text{exec}} = \begin{cases} t^{\text{GCS}}_{\text{process}} + 2t_{\text{prop}} + t_{\text{trans}} + t_{\text{fb}}; & \text{executed at GCS} \\ t^{\text{UAV}}_{\text{process}}; & \text{executed at UAV} \end{cases}$$

Delayed-reward State-Action-Reward-State-Action (SARSA) Framework

- SARSA is well-suited compared to DL models for the small action set in offloading-type problems
- Described by a 4- tuples $\langle S, A, T, R \rangle$
- S: set of all, states the agent can encounter,
  A: set of all available actions,
  $T(s_{n+1}, a_n, s_n) = P(s_{n+1}|s_n, a_n)$: transition function, and
  R: set of all rewards
- Two networks: current network and target network
- Current network Q-table updated as soon as a delayed reward is received; target network Q-table updated periodically
- **State space**: $\{$Critical, Non-critical$\} \times \{$Good channel, Bad channel$\}$
  - s1: {Good channel, Non-Critical frame}
  - s2: {Good channel, Critical frame}
  - s3: {Bad channel, Non-Critical frame}
  - s4: {Bad channel, Critical frame}
- UAV requires minimum N good time slots, out of $N' = \lceil T_{\text{wait}} / T_{Coh} \rceil$ slots required to successfully transmit a frame to GCS
- State of the n-th frame and its channel condition are predicted using 2-state MDPs
- Depending on the state, the agent takes action $a_n \in A$ and receives a delayed reward r_n using target network

- **Action space**: two possible actions $a_n \in A$ at UAV for the n-th frame
  - $a_n = 1$: Send to GCS immediately and wait for feedback for a maximum of T_{wait} duration
  - $a_n = 2$: Process at UAV
- Optimization problem to determine the optimal offloading policy is

$$\begin{aligned} \max_{\pi} \quad & f(E_{\text{rem}}) \\ \text{s.t.} \quad & C1: 0 \le E_{\text{rem}} \le E_{\text{max}} \\ & C2: D_n^C \le D_{th}^C; \\ & C3: D_n^{NC} \le D_{th}^{NC}, \ \forall n \end{aligned} \tag{3}$$

  - E_{rem}: remaining UAV energy

$$E_{\text{Rem},n} = \begin{cases} E_{\text{Rem},n-1} - t_{\text{trans}} P_t; & a_n = 1 \\ E_{\text{Rem},n-1} - t_{\text{process}}^{\text{UAV}} P_{\text{process}}^{\text{UAV}}; & a_n = 2 \end{cases} \tag{4}$$

  - D_n^C: critical frames drop rate, D_n^{NC}: non-critical frames drop rate
  - D_{th}^C: threshold limit of D_n^C, D_{th}^{NC}: threshold limit of D_n^{NC}

$$D_n^{C/NC} = \frac{\text{No. of critical/non-critical frames dropped}}{\text{Total delayed feedbacks available}} \times 100\% \tag{5}$$

Rewards

- To solve (3), we use reward shaping technique to directly incorporate the constraints
- Total reward over all the N_F frames of a video is

$$\text{Reward} = \frac{1}{N_F}\sum_{n=1}^{N_F} \eta_1 r_{n,1} + \eta_2 r_{n,2} + \eta_3 r_{n,3} \tag{6}$$

- η_1, η_2, and η_3 are indicator variables
- $r_{n,1}$: reward/penalty for the judicious use of remaining UAV energy

$$r_{n,1} = \frac{2}{(1+e^{b_1 \frac{E_{\text{Rem},n}}{E_{\max}}})} \in [0,1] \tag{7}$$

- Reward functions corresponding to the critical frame drop constraint and non-critical frame drop constraint are

$$\begin{aligned} r_{n,2} &= z_n(D_n^C, D_{th}^C) \in [-1,1]; \\ r_{n,3} &= z_n(D_n^{NC}, D_{th}^{NC}) \in [-1,1]. \end{aligned} \tag{8}$$

- Where

$$z_n(D_n, D_{th}) = \begin{cases} \left(\frac{E_{\text{rem},n}}{E_{\max}}\right)^{c_1} \left(1 - \frac{2}{\left(1+e^{b_2 \frac{D_n - D_{th}}{D_{th}}}\right)}\right); & D_{th} > 0 \\ c_2 \left(\frac{E_{\text{rem},n}}{E_{\max}}\right)^{c_1}; & D_{th} = 0 \end{cases} \tag{9}$$

State-Action-Reward design

{Channel during transmission of n-th frame, n-th frame semantics}	Action a_n	Frame drop rate conditions	Reward indicators
{Good, Non-critical}	a_n=1	$D_n^{NC} \leq D_{th}^{NC}$	$\eta_1=+1, \eta_2=0, \eta_3=+1$
		$D_n^{NC} > D_{th}^{NC}$	$\eta_1=+1, \eta_2=0, \eta_3=-1$
	$a_n = 2$	$D_n^{NC} \leq D_{th}^{NC}$	$\eta_1=-1, \eta_2=0, \eta_3=+1$
		$D_n^{NC} > D_{th}^{NC}$	$\eta_1=-1, \eta_2=0, \eta_3=-1$
{Good, Critical}	$a_n = 1$	$D_n^{C} \leq D_{th}^{C}$	$\eta_1=+1, \eta_2=+1, \eta_3=0$
		$D_n^{C} > D_{th}^{C}$	$\eta_1=+1, \eta_2=-1, \eta_3=0$
	$a_n = 2$	$D_n^{C} \leq D_{th}^{C}$	$\eta_1=-1, \eta_2=+1, \eta_3=0$
		$D_n^{C} > D_{th}^{C}$	$\eta_1=-1, \eta_2=-1, \eta_3=0$
{Bad, Non-critical}	$a_n = 1$	$D_n^{NC} \leq D_{th}^{NC}$	$\eta_1=+1, \eta_2=0, \eta_3=-1$
		$D_n^{NC} > D_{th}^{NC}$	$\eta_1=-1, \eta_2=0, \eta_3=+1$
	$a_n = 2$	$D_n^{NC} \leq D_{th}^{NC}$	$\eta_1=-1, \eta_2=0, \eta_3=+1$
		$D_n^{NC} > D_{th}^{NC}$	$\eta_1=+1, \eta_2=0, \eta_3=-1$
{Bad, Critical}	$a_n = 1$	$D_n^{C} \leq D_{th}^{C}$	$\eta_1=+1, \eta_2=-1, \eta_3=0$
		$D_n^{C} > D_{th}^{C}$	$\eta_1=-1, \eta_2=+1, \eta_3=0$
	$a_n = 2$	$D_n^{C} \leq D_{th}^{C}$	$\eta_1=-1, \eta_2=+1, \eta_3=0$
		$D_n^{C} > D_{th}^{C}$	$\eta_1=+1, \eta_2=-1, \eta_3=0$

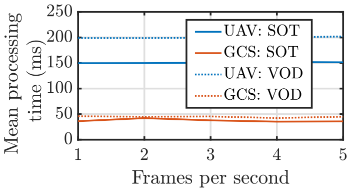
Experimental Results: Mean processing time and Mean Power consumption

Table 1: Object detection and tracking algorithm specifications

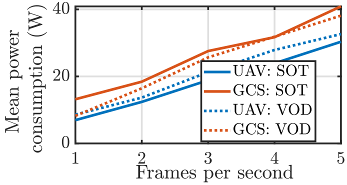
Algorithm	**Video object Detection (VOD)**	**Single Object Tracking (SOT)**
Model	SiamRPN++	Deep feature flow
Backbone	R-50	R-50-DC5
Training dataset	ImageNET VID	LaSOT
Accuracy	50.4 Success	70.3 box AP@50

Table 2: UAV and GCS server specifications

Server	**Server 1**	**Server 2**
Class	UAV	GCS
CPU	Intel i7-7700HQ	2× AMD EPYC 7601
CPU TDP (W)	45	2 × 180
GPU	Nvidia MX150	Nvidia GV100
GPU TDP (W)	25	250



(a) Mean processing time t_{process}



(b) Mean power consumption P_{process}

Figure 5: Mean processing time t_{process} (a) and mean power consumption P_{process} (b) of the SOT and VOD algorithms, when implemented on UAV server and GCS server.

Powerful server at GCS reduces processing time of both SOT and VOD algorithms compared to local processing on a small server onboard the UAV at the cost of higher computational power while maintaining an accuracy of 70.3%!

Simulation parameters

System parameters

- Carrier frequency $=$ 28 GHz; Bandwidth $=$ 400 MHz
- Mean bits per frame $=$ 10 MB
- T_{Coh}=10 ms; T_{wait}=200 ms
- t_{trans}=25 ms with 1 bps rate; also, t_{prop}=0.0003 ms
- Assuming feedback packet of 1 KB, t_{fb}=0.0025 ms; N=8

From the experimental results, at FPS=5 of SOT algorithm

- UAV frame set processing time $t_{\text{process}}^{\text{UAV}}$=151.3 ms
- GCS frame set processing time $t_{\text{process}}^{\text{GCS}}$=35.72 ms
- Mean power required for running the SOT algorithm at the UAV =40.89 W
- Mean UAV transmission power $P_t = 1$ W

Other simulation parameters

- Frame transition probability matrix $= [0.85, 0.15; 0.15, 0.85]$
- Channel transition probability matrix $= [0.9, 0.1; 0.2, 0.8]$
- D_{th}^{C}=0%; D_{th}^{NC}=5%
- b_1=3; b_2=3
- c_1=0.01; c_2=0.8
- Learning rate α=0.2; Discount factor γ=0.4; $\epsilon = 0.2$ with ϵ-decay factor of 0.995

Results of proposed RL framework

Compared our proposed 4-states RL framework with the following:

- *1-state RL*: only 1 RL state, and action $a_n = 1$ or $a_n = 2$;
- *2-states RL*: 2 RL states $S' = \{\text{Good channel, Bad channel}\}$ and action action $a_n = 1$ or $a_n = 2$

Processing all frames locally on UAV consumes maximum UAV energy $E_{\max}$=4.9505 KJ, while offloading all frames to the GCS incurs 0.025 KJ of UAV energy

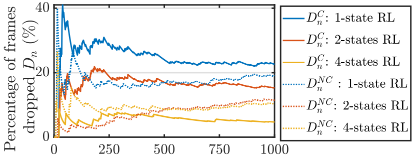


Figure 6: Frame Drop rates $D_n^{C/NC}$

- Frame criticality statistics improve D_n^C and D_n^{NC} over using just channel statistics for UAV training
- Not incorporating channel or frame semantics statistics to design state causes severe packet loss, indicating overfitting
- D_n^C and D_n^{NC} values fail to meet required thresholds as constraints have been designed using reward functions that either reward or penalize the system

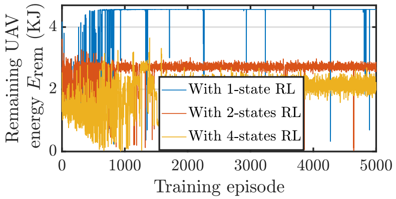


Figure 7: Remaining UAV energy E_{rem}

- 4-state RL consumes more energy but has smaller frame drop rates comapred to 2-state RL
- With 1-state RL, all the frames offloaded to GCS and have high frame drop rate

Conclusion

- A power-intensive object learning algorithm does not need to run on every frame $\Rightarrow$ power savings for the UAV $\Rightarrow$ prolonging its operation
- Energy consumption of the UAV, utilizing reinforcement learning, lies between the two extremes of transmitting all frames to GCS for processing and processing all frames locally on the UAV
- Important to consider the communication channel condition when deciding to offload frames to the GCS along with frame criticality statistics
- Proposed 4-states RL framework stands out as it achieves the lowest critical frame drop rate of 5% while saving approximately 45% of UAV energy

Thank You !

E-mail: Nancy.Varshney@dbst.iitd.ac.in, {corrado.puligheddu, carla.chiasserini, claudio.casetti}@polito.it, and swadesd@ee.iitd.ac.in