DynaaDCP: Dynamic Navigation of Autonomous Agents for Distributed Capture Processing

Sam Jijina, Ramyad Hadidi, Jun Chen, Zhen Jiang, Ashutosh Dhekne, Hyesoon Kim
{sam.jijina, rhadidi, jchen706, zjiang330, dhekne, hyesoon.kim}@gatech.edu
Georgia Institute of Technology

1 INTRODUCTION

Effective autonomous agents must perform complex tasks and analyze multidimensional raw data to function in diverse situations. Even though all the operations performed on these agents require computation power, with the advancement of deep learning and computer-vision algorithms, analyzing the multidimensional raw data constitutes a large portion of these computations. Specifically, due to the limited energy and computation power of individual autonomous agents (*e.g.,* drones), it is a challenge to perform all the computations on a single agent. One solution to this challenge is integrating high-performance computing units such as GPUs or customized ASICs. However, such an approach will incur high energy costs which will add extra battery weight. Another solution is to offload computations to the cloud or a nearby base-station, necessitating high-speed data connections to powerful datacenters. Nevertheless, this solution requires additional costly datacenter machines and dedicated low-latency and high-bandwidth links to autonomous agents, which are almost impossible to acquire due to the agent's movements and ever-changing environments, and potentially remote operations.

In this paper we present DynaaDCP, a novel and yet generic framework to suggest a *mobility pattern* for autonomous agents so that distributed computation on these agents benefit from reduced communication overhead. We demonstrate this capability through a set of exemplar applications of computer vision algorithms running in parallel on multiple autonomous agents. Concretely, we show how a set of autonomous survey drones capturing and processing aerial images benefit from this approach. The drones, while using distributed computer vision algorithms, alternate between getting close to each other and moving back to their survey locations to improve the overall computation efficiency.

2 SYSTEM ARCHITECTURE AND DESIGN

Figure 1 illustrates two critical configurations in our DynaaDCP framework using autonomous drones capturing and processing aerial images as an example. The two configurations are as follows: (i) drones are far away from each other and capturing aerial images while performing computation on the previous batch, and (ii) drones are near each other to communicate recently acquired aerial images using high data rate links. Processing captured data in-flight allows utilizing the compute power of the drones to the fullest, while also enabling dynamic mid-flight decision making. Our focus in this work is on improving communication efficiency which in turn improves the overall efficiency of the distributed algorithms. Existing studies categorize the communication link between processing units in two categories: (1) multi-processor distributed systems that assume a tight linkage between various processing units, and therefore, a low communication overhead; and

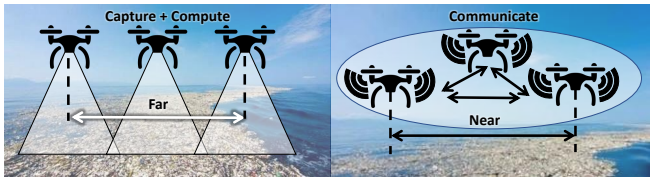

Figure 1: DynaaDCP near-far schedule guides autonomous agents to alternate between communication and capture tasks, minimizing the overall time needed for completing a mission.

(2) geo-distributed systems, which in contrast, are built assuming a significant uncertainty about the communication overhead.

We propose that there is a *third category* of communication links, that is a distributed system in which we have control over the parameters of the communication links. For example, if a fleet of flying drones comes within physical proximity of each other (reducing the inter-drone distance), then higher communication data rates are achievable. Furthermore, provisioning more than one communication technology, such as Wi-Fi and mmWave [3], allows switching between long-range and high data-rate alternatives when appropriate. If drones are organized such that they are within a few meters of each other, then mmWave links provide high throughput between the drones, enabling extremely fast exchange of information which is then made available to the computing units even when the drones are far away.

We envision two design settings in which we expect to use DynaaDCP: (1) pre-programmed mode, and (2) reactive mode. For any application or workload, we expect that the reactive mode is always be applicable. For many applications with fixed mission objectives, the pre-programmed mode should be applicable, which provides the most benefit from innovations proposed in DynaaDCP.

■ **Pre-programmed Mode:** The drones will be pre-programmed with a specific mission that comprises several runs of a set of sub-tasks. Each run, shown in Figure 2, accounts for many aspects of the drones' flight. At the end of each run, the drones can make future path planning decisions based on the results of the previous computations. To this end, DynaaDCP develops an analytical algorithm to deduce a movement schedule for a set of autonomous agents given a certain expected computation and communication trace, illustrated in Figure 3. In the following, we describe the design of our system in detail.

■ **Reactive Mode:** In this mode, we relax our assumption of complete knowledge of the duration of each sub-task. Instead, we take a simpler approach that should be applicable to all workloads. We just assume we know the duration of the communication task before the communication begins. Thus, in the reactive mode, DynaaDCP decisions are limited to modified behavior during the communicate subtasks only.

Sam Jijina, Ramyad Hadidi, Jun Chen, Zhen Jiang, Ashutosh Dhekne, Hyesoon Kim
{sam.jijina, rhadidi, jchen706, zjiang330, dhekne, hyesoon.kim}@gatech.edu
Georgia Institute of Technology

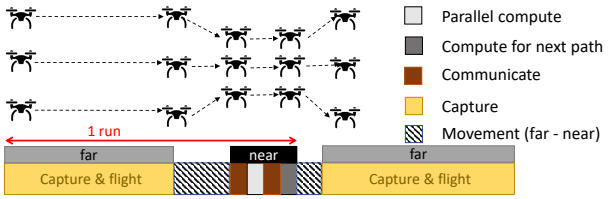

Figure 2: Overview of a mission run executed by autonomous agents. Each mission is comprised of multiple runs, and each run has multiple compute, communicate, and capture sub-tasks.

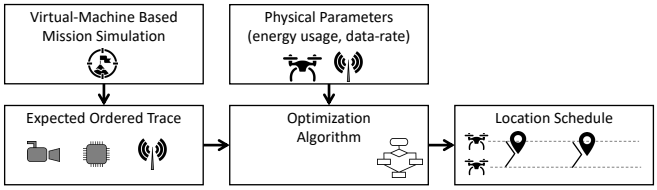

Figure 3: Envisioned System Design for the pre-programmed mode.

3 ALGORITHM DEVELOPMENT

At a high level, DynaaDCP abstracts out the specifics of the distributed task that the user wishes to run on a set of autonomous agents. So, whether a set of drones are running a graphics depth map, or a set of self-driving all-terrain vehicles are mapping a surface, DynaaDCP's abstraction allows simply dividing the entire complex end-task into a sequence of capture, compute, and communicate subtasks. Among these, only the communication subtask is elastic; we expect the trace to contain the approximate *amount of data* to be transferred, and our algorithm converts it to *time elapsed* based on the expected data transfer rate achievable. Other subtasks can be represented in durations of time, though "number of operations", and "number of captured entites" are also acceptable ways to represent the compute and capture subtasks.

The bottomline for the DynaaDCP algorithm is that it must decide between two possible *locations* for the autonomous agents at every time instant—*near* or *far*. **A globally optimal algorithm that** takes an expected trace as input and **produces a "near-far" movement schedule** with the aim of completing the set task in the minimum amount of time or with the minimum amount of energy consumed, **is our primary contribution in this paper.**

4 RESULTS

Distributed Computer-Vision Processing: We first analyze our distributed computer vision processing on a Raspberry Pi 4 with configurations of 2 GB of RAM with unthrottled bandwidth of around 940 *Mbps*. The distributed pipeline for generation of `orthophoto` includes EXIF extractions, detect features and feature matching among the images, cluster computation of OpenSfm's structure from motion [2], MVE's multi-view stereo [1], and OpenDroneMap's orthophoto generation [4]. The comparison tradeoff results are shown in Figure 4.

4.1 Effectiveness of DynaaDCP

We evaluate DynaaDCP's performance with respect to the flight time and the energy saved per run. We execute the DynaaDCP algorithm over multiple traces of 6 computer-vision missions: computing a stitched orthophoto from a group of images, 2 traces generating depth maps, and 3 traces with substantial time spent in movement, capture, and communication respectively, for a total of

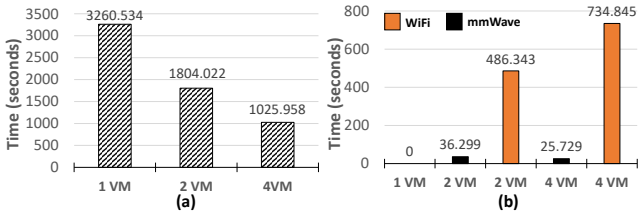

Figure 4: Distributed Computer Vision Processing on VMs: (a) Compute Benchmarks, (b) Communication Benchmarks.

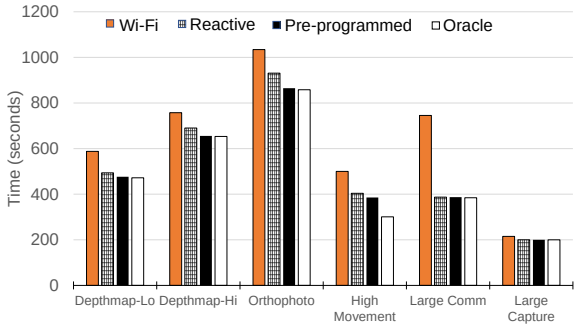

Figure 5: DynaaDCP reduces the total flight time required to complete the mission, can comes close to the hypothetical least time required (Oracle).

6 traces. For each trace, we compare DynaaDCP performance with the performance expected when only Wi-Fi is used by all drones, as a lower bound.

As an upper-bound, we compare DynaaDCP's performance with a *hypothetical oracle* that allows a single communicate subtask to be broken into Wi-Fi and mmWave parts, allowing the best communication paradigm when transiting between `near` and `far` locations. These comparisons are summarized in Figure 5, which shows that a significant speedup is offered by DynaaDCP over the baseline, while not being too far from the hypothetical Oracle's performance.

5 CONCLUSION

Mobility of autonomous agents poses significant challenges to the execution of collaborative distributed algorithms. However, in this work, we showed that if we can control the motion patterns of the autonomous agents, mobility can be used to selectively improve communication data rates when demanded by the distributed algorithm. We believe DynaaDCP provides a new direction to research in this area with mobility providing an additional tuning knob in the distributed systems.

ACKNOWLEDGMENTS

We thank the DOSSA program and review committee for their valuable feedback in improving our paper. We gratefully acknowledge the support of NSF OAC 2103951.

REFERENCES

[1] Simon Fuhrmann, Fabian Langguth, and Michael Goesele. 2014. MVE-A Multi-View Reconstruction Environment.. In *GCH*. Citeseer, 11–18.
[2] Mapillary. 2020. OpenSfM. https://github.com/mapillary/OpenSfM
[3] Thomas et al. Nitsche. 2014. IEEE 802.11 ad: directional 60 GHz communication for multi-Gigabit-per-second Wi-Fi. *IEEE Communications Magazine* 52, 12 (2014), 132–141.
[4] OpenDroneMap. 2020. ODM. Retrieved Mar 1, 2021 from https://github.com/OpenDroneMap/ODM