Autonomous Methods in Multisensor Architecture for Smart Surveillance

Dani Manjah[1][a], Stéphane Galland[2][b], Christophe De Vleeschouwer[1] and Benoît Macq[1][c]
[1]*Institute of Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), UCLouvain, 1348 Louvain-la-Neuve, Belgium*
[2]*UTBM, CIAD UMR 7533, F-90010 Belfort cedex, France*

Keywords: Distributed Smart Cameras Architectures, Holonic Multiagent Systems, Large-Scale Surveillance Systems, Multisensor and Multimethod.

Abstract: This paper considers the deployment of flexible and high-performance surveillance systems. These systems must continuously integrate new sensors and sensing algorithms, which are autonomous (e.g., capable of making decisions independently of a central system) and possess interaction skills (e.g., capable of exchanging observations). For this purpose, our work proposes adopting an agent-based architecture derived from an organizational and holonic (i.e., system of systems) multi-agent model. It leverages autonomous processing methods, resulting in a scalable and modular multisensor and multimethod surveillance systems. A vehicle tracking case study demonstrates the relevance of our approach in terms of effectiveness and runtime.

1 INTRODUCTION

Surveillance systems traditionally employ a multi-sensor architecture, enhancing system survivability against device defects or temporary occlusions (Rao et al., 1993). Despite this, prevailing systems predominantly rely on singular, unique processing methods (Wolpert and Macready, 1997), limiting adaptability and flexibility in the face of evolving contexts and scenarios. Next-generation surveillance systems should consist of networks with interconnected sensors and methods endowed with reasoning (i.e., information processing) and interaction skills. Although existing architectures enable the management of complex structures of sensors, they assume that methods are designed beforehand, thereby reducing the range of applications and system flexibility.

Our paper addresses the design of a scalable, modular, and adaptive multisensor and multimethod surveillance system. Our approach involves breaking down the system into smaller, manageable parts like sensors and methods. Each part can operate on its own, making decisions without needing approval from a central authority. Also, each part can communicate, asking for and sharing information as needed.

Conventional software engineering techniques, like UML, often fall short in managing the complexity (Simon, 1976) of large-scale multi-sensor and multi-method systems due to their foundational design concepts (Rombach et al., 1993; Abbas, 2015; Wautelet et al., 2021). In contrast, our work embraces the concept of Multi-Agent Systems (MAS), more particularly leveraging the capabilities of Holonic Multi-Agent Systems (HMAS). A HMAS refers to a multi-agent system in which each agent is assigned to a self-similar nesting structure, known as a holon. HMAS represents a promising approach for complex system modeling due to its capability to support the key properties of a complex system, including flexibility, intelligence, and scalability. This approach allows for a more organic and responsive system architecture, akin to living organisms, promoting autonomy and dynamic interactions among system components. Additionally, due to the stochastic interactions within the real world, the *agent paradigm* must be augmented with *organizational concepts* (actors, goals, objectives, responsibilities, social dependencies, etc.) that provide abstract patterns of interaction, ensuring model scalability and flexibility (Ferber et al., 2003; Cossentino et al., 2010; Abbas, 2015).

This paper focuses on designing an architecture that facilitates autonomous Sensor Management and

[a] https://orcid.org/0000-0001-9034-0794
[b] https://orcid.org/0000-0002-1559-7861
[c] https://orcid.org/0000-0002-7243-4778

Manjah, D., Galland, S., De Vleeschouwer, C. and Macq, B.
Autonomous Methods in Multisensor Architecture for Smart Surveillance.
DOI: 10.5220/0012395700003636
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 3*, pages 823-831
ISBN: 978-989-758-680-4; ISSN: 2184-433X
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

promotes Autonomous Methods for enhanced system performance. To the best of our knowledge, there exists no model that guarantees such functionalities in both sensors and methods concurrently. The contributions of this paper are:

1. **Design of an Organizational Holonic Agent Architecture:** accommodating multi-sensor and multi-method systems.
2. **Implementation:** employing the SARL-agent language (GALLAND et al., 2019). The link for the multi-agent framework is available at https://github.com/manjahdani/aptitude.
3. **Experimental Demonstrations:** with simple behaviors for image-processing algorithms while improving the efficiency of a system.

This paper begins by discussing major related work in smart surveillance systems in Section 2. Next, in Section 3, we describe our system, and in Section 4, we describe the ASPECS methodology (Cossentino et al., 2010). In Section 5, we provide large-scale experiments on 40 videos in *The AI-City challenge* (Naphade et al., 2021) from the using our multi-agent surveillance framework *APTITUDE*. We conclude the paper with future directions and conclusions in Section 6.

2 STATE OF THE ART

2.1 Surveillance Architectures

Traditionally, distributed surveillance systems are designed as centralized systems with a topology configuration in which the sensors are like the tentacles of a central node (Valencia-Jiménez and Fernández-Caballero, 2006). The simplicity of the approach, its known performances, and controllability come with an **inherent bottleneck** restricting scalability. To improve the scalability of surveillance systems, researchers (Park et al., 2017; Hilal and Basir, 2015) have focused on a multi-layer management of sensors. In this hierarchical approach, the central node decomposes the processing loads into smaller assignments. The advantage is the minimization of communication costs, as only the results are passed on to the upper layers. The approach is suitable if the sensors are homogeneous and setup in a circumscribed environment. In fact, the assumption fails in most real scenarios, where the sensors and situations are heterogeneous. For example, in a video surveillance system that monitors one crossroads, it is common to have various types of camera (*PTZ, Bullet, Dome, etc.*). Furthermore, they can be set at various heights (Elharrouss et al., 2021) and have individual goals. A solution for better adaptability could then be a **decentralized architecture** allowing local control and decision making. Preferably, **a hybrid architecture** would provide interesting alternatives (Mihailescu et al., 2018).

A common issue in the literature is to design a surveillance system as a closed network with static components. This assumption is being increasingly challenged. First, the massive *Internet of Things (IoT)* enables cities to become urban sensing platforms (Perera et al., 2014) and potential users of the system. Furthermore, it may be economically advantageous to employ large numbers of cheaper miniaturized devices (*Jetson Nano, Raspberry Pi, etc.*). Naturally, this increases the probability that sensors will arrive and leave or fail during the processes. Systems will thus be increasingly open in the future. This means that nodes constantly need to negotiate and self-adapt to ensure the best coverage (Saffre and Hildmann, 2021). In (Hilal and Basir, 2015), the authors designed sensors as intelligent *agents* (autonomous, social, proactive and reactive nodes), using the framework of *Belief, Desire and Intention* (Wooldridge and Jennings, 1995). This agent-oriented paradigm allows **active** rather than passive cooperation of sensors. We claim it designs best next-generation surveillance systems.

Another common point in the literature (Park et al., 2017; Hilal and Basir, 2015; Chen et al., 2009; Valencia-Jiménez and Fernández-Caballero, 2006) is that it falls within the scope of *sensor management* and set-up experiments on simulated data or in small-scale test-bed cases. However, to be able to move toward large-scale end-to-end deployments, it is critical that systems' integration aspects be carefully considered (Mihailescu et al., 2018). Too often, methods are given secondary importance in such studies. However, a portfolio of methods brings a new range of possibilities. For instance, an ensemble of methods, processing in parallel, could reduce the predictions' uncertainty. Another example is that the system could self-adapt to select a suitable method for a situation (Figure 1b). In conclusion, in addition to working on real datasets and scenarios, one must distinguish between sensors and methods to increase the model's modularity and broaden the range of applications of a surveillance system.

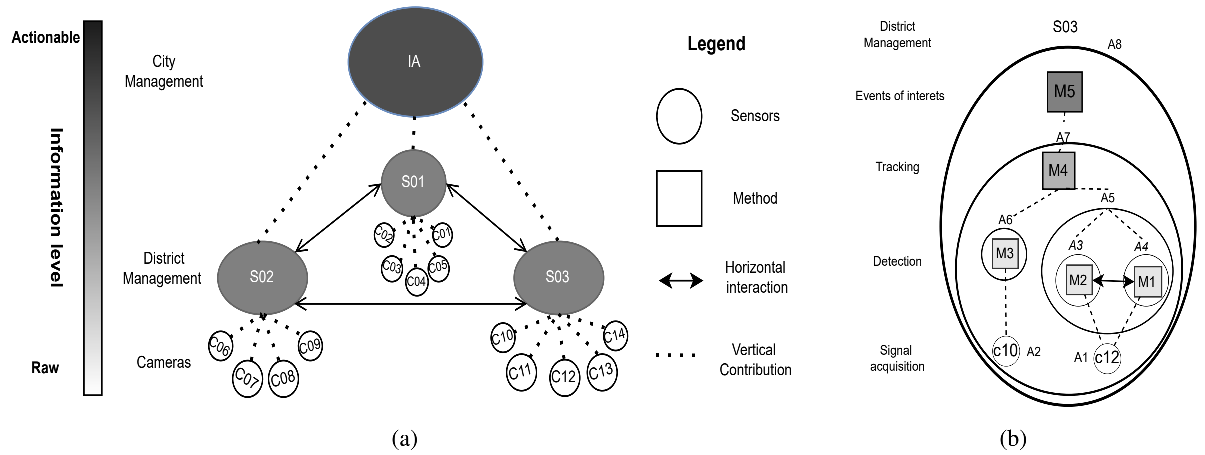

Figure 1: (Left) Figure 1a illustrates the conventional sensor management architecture. (Right) Figure 1b, inspired by Section 5.2, depicts the multi-layer signal processing in applications, highlighting the coordinated roles of Agents A3, A4, and A5, guided by Agent A7. These figures juxtapose the complex dynamics of multisensor and multimethod systems and expose the limitation of traditional sensor-based architectures: their inability to adaptively manage the interactions among multiple methods without pre-established configurations. This contrast draws attention to the need for an evolution towards more adaptive, integrated systems in signal processing, enhancing flexibility and efficiency through method autonomy and enriched interactive capabilities.

2.2 Agent-Oriented Software-Engineering

Large-scale multi-sensor and multi-method systems are complex (Simon, 1976). Conventional software engineering techniques, like UML, offer tools limited by their foundational design concepts, like procedural scripting (Abbas, 2015; Wautelet et al., 2021). Consequently, they do not facilitate managing systems that aim to expand both vertically (in control and information layers) and horizontally (in physical distribution) as illustrated in Figures 1a to 1b. Engineering should instead mimic living systems and their organization, endowing software components with autonomy and interaction abilities (Rombach et al., 1993; Abbas, 2015; Wautelet et al., 2021). We refer to this concept as the *“agent”* paradigm.

Multi-Agent Systems (MAS) are considered a pertinent paradigm for modeling complex systems, comprising connected autonomous agents working towards a common goal in a specific environment (Rodriguez et al., 2011). Traditional MAS models view agents as atomic entities, limiting their application in exhibiting hierarchical structure. Holonic Multi-Agent Systems (HMAS), however, overcome this limitation by utilizing ’holons’—self-similar entities representing whole-part relationships, enabling detailed and modular modeling of large-scale complex systems through aggregation of nested agent accomplishments (Gerber et al., 1999; Rodriguez et al., 2011). Finally, the stochastic interactions within the real world imply that the *agent paradigm* must be augmented with *organizational concepts* (actors, goals, objectives, responsibilities, social dependencies, etc.) that provide an abstract pattern of interaction, thereby ensuring model scalability and flexibility (Ferber et al., 2003; Cossentino et al., 2010; Abbas, 2015). The motivation of an organizational approach to modeling complex systems is further elaborated upon in (Ferber et al., 2003; Cossentino et al., 2010) and formally included in the ASPECS methodology (Rodriguez et al., 2011). It specifies the problem to be solved using an organizational modeling approach based on the CRIO metamodel and designs a solution based on HMAS. The problem specification for surveillance systems is detailed in Section 4.1. Section 4.2 provides the mapping to the HMAS.

3 PROBLEM STATEMENT

Multisensor and multimethod surveillance systems involve large-scale networks of *interconnected* components (e.g., sensors and methods) endowed with *reasoning* (i.e., information processing) and *interaction* skills. We refer to the aforementioned concept as an agent (Wooldridge and Jennings, 1995). They collaborate to accomplish a given objective. For instance, in our case study of traffic analysis, multi-camera cooperation is needed for city-scale tracking. The engineering of this particularly complex system (Simon, 1976) **should meet three main specifications**:

C1. Scalability and modularity. A complex surveillance system should be able to continuously integrate more and more sensors (Valencia-Jiménez and Fernández-Caballero, 2006) and new methods. It could itself play a role in a higher-order surveillance system. In other words, the system should be *scalable* and *modular*; the integration of new and heterogeneous sensing methods in particular should be facilitated.

C2. *Resilience.* We describe the system's ability to gracefully degrade and recover under stress or when a node is removed. As a general principle, systems employing multiple sensors and methods exhibit greater fault tolerance. For instance, (Liu et al., 2020) demonstrates how cameras observing the same scene can compensate for a malfunctioning camera (caused by occlusion or fault) by reconstructing missed detections. Regarding methods, employing an ensemble approach reduces false positive/negative rates and anomalies.

C3. *Openness.* Traditionally systems are mostly designed as closed networks. This design is increasingly being challenged. The massive Internet of Things (IoT) sensors have enabled cities to become massive sensing platforms (Perera et al., 2014). Furthermore, economically, it may be advantageous to employ large numbers of cheaper, miniaturized devices (*Jetson Nano, Raspberry Pi, etc.*), which naturally increases the probability of agents arriving, leaving or failing during the processes. Preliminary studies have shown that ignoring this need for openness may lead to severe losses of performance(Abdelrahim et al., 2018; Hendrickx and Martin, 2018). The management of *openness* implies self-organisation capabilities, such that the system can self-adapt if a component enters or leaves the system.

Conventional systems generally struggle to fully meet those specifications. Sensor management architectures with a topology similar to Fig. 1a can handle scenarios involving multiple cameras observing the same scene (Fig. 2) or city-scale tracking where the smart sensors select the appropriate method for each camera (Fig. 3). Nevertheless, their structure assumes that the processing methods are designed beforehand as a sequence of intertwined processes. For instance in a tracking by detection, the tracker could be programmed to receive multiple detections. However the integration of new behaviours (e.g., voting mechanisms in case of uncertainty) requires to adapt the tracker's module for each camera in which we want to install the behaviour. In contrast, if each component (tracker and detectors) is designed as an autonomous self-adapting block, deployment of resilient and flexible system behaviours can be facilitated.

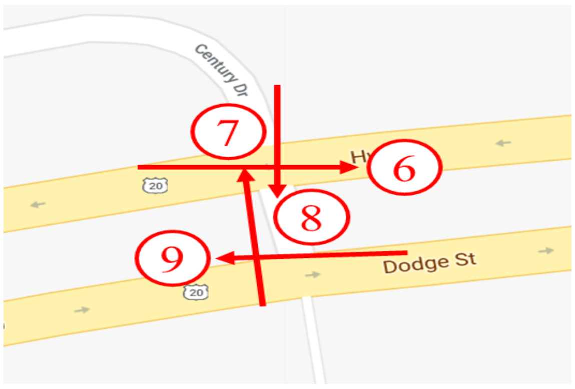

Figure 2: District 2. A network of cameras observing the same target bolsters the system's resilience as the failure (e.g., fault or occlusions) of one or several cameras can be balanced by the other functioning cameras.

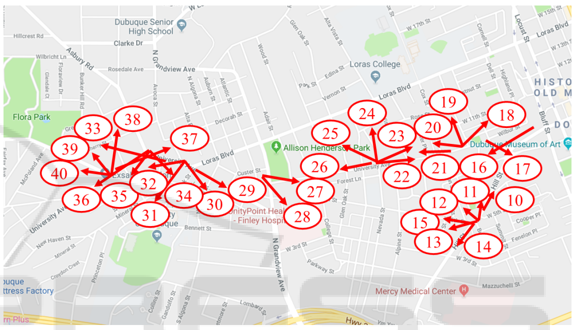

Figure 3: Positioning of the cameras in the case study of Section 5.1. A general-purpose method produces underperforming systems. There is no dataset large enough to cover all aspects of a city scenario (Soviany et al., 2021).

4 HOLONIC ARCHITECTURE

As a preliminary point, the concepts of ***organization*** and ***role*** must be defined. An ***Organization*** represents a subsystem in which entities play roles and interact together to achieve a shared goal in the context of this organization. A ***Role*** is both an expected behavior to fulfill (part of) a requirement and a status to the role's player in the organization. At the agent level, they can select one or more roles to be played by them and execute the corresponding behaviors.

Holarchy is the hierarchical structure composed of the agents/holons of the system. Each level of this holarchy is a possible instance of an organization that is defined in the problem specification (as illustrated in Section 4.1). Each holon plays one or more roles in one or more groups (instances of organizations) on the same level. Composition relations between super-holons and sub-holons are then specified in accordance with the contributions from the organizations defined in the organizational hierarchy.

4.1 Organizational Model

The section presents the organization *cyber-physical platform* (CPP) (Fig. 4) and its roles ***Resource Provider***, ***Observer*** and ***Sensor***.

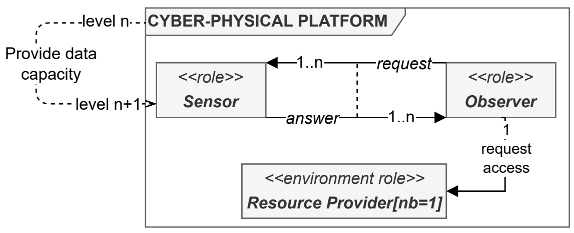

Figure 4: Organizational model of the *cyber-physical platform*, using the ASPECS notation (Cossentino et al., 2010). The ***Resource Provider*** role ensures a fair distribution of the resources among all parties. The ***Observer*** role has the capacity to deliver perceptions thanks to the data acquired by the ***Sensor*** role. The acquisition of data could rely on another CPP.

CPP has the capacity to provide perceptions to external requests. It has finite resources while ensuring fair access to multiple surveillance operations. Therefore, the ***Resource Provider*** role handles authorizations of legitimate interactions and management of resources. For instance, it could prevent the execution of a mission requiring more resources than available (e.g., prevent the fusion of several methods) or the use of a method disrespectful of a standard (i.e., privacy). The ***Observer*** role has the capacity to provide perceptions after processing data given by sensors. Multiple ***Observers*** can operate in parallel within the organization in collaboratively or competitively. The behaviour of the ***Sensor*** role is to acquire data. The acquisition of data could rely on another CPP. A pseudo-code of the behaviours is provided in the Algorithm 1 in the APPENDIX section.

4.2 Holarchy Design

A holarchy is represented in Fig. 5. The level n represents an agent providing perceptions to any requester. At level $n-1$, holons with ***Observer*** role (e.g., H1 and H2) collaborate (or negotiate) to reach a common objective (e.g., counting vehicles or pedestrians). Each holon with an ***Observer*** role implements a specific method or algorithm to analyze the input of a holon that endorses the ***Sensor*** role (e.g., H3). Upon disagreement, they can negotiate a consensus (e.g., agreement on the type of object). The level $n-2$ shows a holon (i.e., H3) that is providing input data. The latter is composed of one or multiple agents (e.g., H4 and H5). That agent can be activated to fulfill H3's objective. The sub-holons could be a network of cameras observing the same scene or different methods processing a signal. In Section 5, we show the time gained thanks to the cooperation between two holons taking turns to process a video stream with varying traffic density.

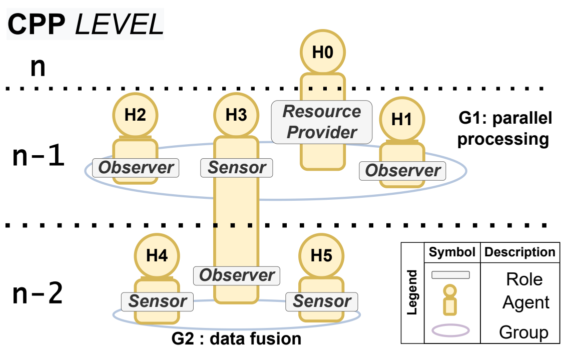

Figure 5: Holonic architecture. The representation of the holarchy was inspired by the "cheese board"notation proposed by (Cossentino et al., 2010; Feraud and Galland, 2017).

5 EXPERIMENTATION

5.1 Case Study

Smart cities continuously face increasing mobility challenges due to ever-growing populations, novel transportation modes (micro-mobility, autonomous cars, etc.) and threats (loitering, abandoned objects, etc.). A resilient and scalable surveillance framework offers cities data-driven governance and could facilitate the advent of connected devices such as wearable blind navigation. City-scale vehicle tracking consists in tracking vehicles across multiple cameras both at a single intersection and across multiple intersections scattered over a city. It helps traffic engineers understand journey times along entire corridors. It can be used to design better intersection signal timing plans and apply other traffic congestion mitigation strategies when necessary (Naphade et al., 2021). The deployment of such large-scale solutions requires both *effectiveness* in tracking and *efficiency* in processing. Traditionally, a surveillance company chooses and deploys one tracker and one detection model across all its solutions. The choice and development of such algorithms are based on their prediction capacity to outperform other existing methods. Yet, the deployment of a unique solution neglects the diversity of contexts in a city (density, lighting, etc.). Consequently, one has to compromise either by deploying an effective but energy-consuming method or a low-cost but poorly effective algorithms. For instance, in

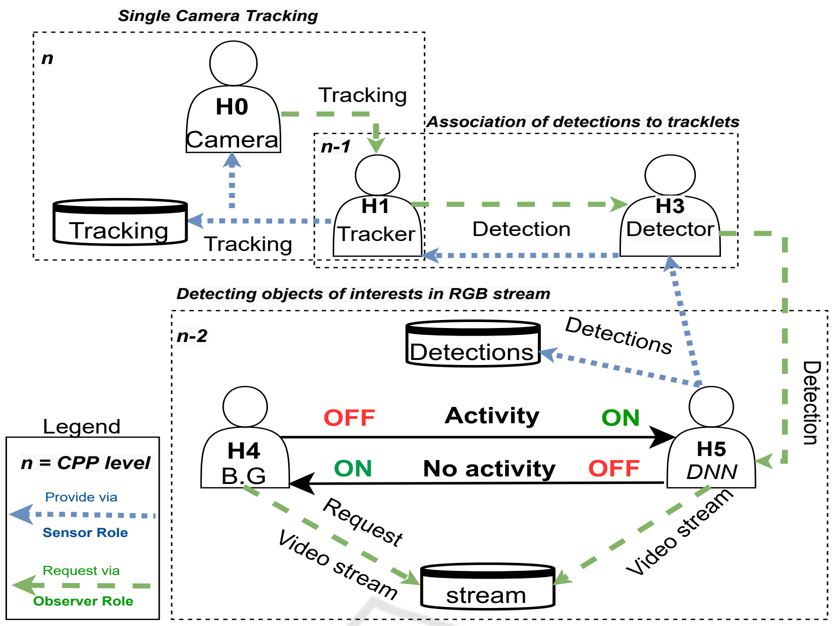

Figure 6: Depiction of Agent Interactions: Camera holon C_i (H0) recruits a tracker T_i (H1) and a detector D_i (H3). Detector D_i engages two agents: an inter-frame differencing agent D_{i1} (H4) and a deep neural network (DNN) agent D_{i2} (H5). H0's tracking request is initially directed to H1, then to H3, which may involve H4, H5, or both. In multi-method operations, H4 notifies H5 of frame activity, leading to H5's engagement as required.

low-density traffic (i.e., fewer objects to track), simple methods (e.g., inter-frame difference) give satisfactory results for detecting the presence of moving objects. Conversely, in high-density traffic (i.e., many vehicles to track), the use of sophisticated methods (e.g., deep neural network) must be used to achieve good tracking performance. The following experiment demonstrates how the collaboration of active and autonomous methods uses fewer resources while maintaining good predictive results for vehicle tracking.

5.2 Practical Design of Agents

The following example illustrates the chain of processes occurring upon the decision of a camera to track vehicles. In this scenario, the tracker estimates the number of tracks as low and informs the detectors that the camera might be in a low-density area (i.e., few objects to track). The detector adopts a strategy that involves recruiting a method (e.g., inter-frame differencing) that assumes the role of an *activity detector*. When movement is detected, it alerts a deep neural network (DNN) that is present but in a saving mode. The advantage of this approach is that the DNN becomes active only when necessary, leading to reduced energy consumption while maintaining effective detection performance. We present these interactions in Fig. 6, and an abstract behavior is provided in the Algorithm 2, in the APPENDIX section.

5.3 Evaluation Protocol

We evaluated the *sensors and methods* architecture's cost at scale against the *single sensor* approach, using 40 cameras from the AI-City challenge dataset (Naphade et al., 2021).

5.3.1 Metrics

Higher Order Tracking Accuracy (HOTA) (Luiten et al., 2021) is a community standard for evaluating the *effectiveness* of multi-object tracking (MOT). In summary, HOTA $\in$ [0,1] measures how well the trajectories of matching detections align and averages this over all matching detections, while also penalizing detections that do not match (Luiten et al., 2021). *Efficiency* is evaluated by the processing time of each processing phase. In the adaptive approach, we added the cost of agentification (i.e., the cost of agent interactions).

5.3.2 Methods

The detector based on the inter-frame differencing is *Background Substraction* from the computer-vision library OpenCV and the deep neural network is a *YOLO v4* model (Bochkovskiy et al., 2020) trained

Table 1: Comparison of large-scale testing of a multi-sensor architecture vs our architecture that integrates autonomous methods. The camera arrangement is depicted in Fig. 3.

Method	**HOTA**	**Proc. Time (s)**	**Agent Time (s)**	**Total (s)**
multisensor, DNN only	0.361	2327.05 ± 21.41	9.95 ± 0.8	2337
multisensor, multimethod	0.360	2169.40 ± 30.84	12.48 ± 0.85	2181.88
Difference	0.001	-157.65	2.53	-155.12

on the MIO-TCD dataset (Luo et al., 2018). In both cases, the tracker is SORT (Bewley et al., 2016).

5.3.3 Test Environment

Finally, the experiments were performed on a computer with an Intel(R) Core(TM) i7-8700K CPU processor running at 3.70GHz and on Windows 10. YOLO v4 ran on GPU with NVIDIA GeForce GTX 1080 Ti. All **publicly** available source codes, selected videos and methods configurations used in this work are available and detailed at (https://github.com/manjahdani/aptitude).

5.4 Results

Table 1 displays the HOTA scores, processing times, and agentification costs, alongside their respective differences. These tests utilized 40 videos from the cameras in the AI-City Challenge (Naphade et al., 2021), as illustrated in Fig. 3. The processing times mentioned are averages derived from 10 simulations. Analysis of Table 1 indicates that the system maintains consistent tracking performance at scale. It also reveals a 25% increase in agent interactions, but overall consumption was reduced by 7%.

6 CONCLUSION

We overcome the lack of flexibility of existing multi-sensor architecture by adding autonomy to the methods. We propose a system in which each component (sensors, methods) is endowed with autonomy, allowing them to make decisions without referencing a central system, and interaction skills, enabling them to request and provide observations. Given the intrinsic complexity of such a surveillance system, we selected the organizational and holonic multi-agent paradigm for modeling. The different organizations and roles of the system are highlighted (Section 4.1) and described, as well as the interactions and role behaviours (Section 4.2). To validate the proposed model, a city-scale vehicle tracking system consisting of multiple cameras at single or multiple intersections across a city was considered (Section 5). The experiment deployed a system embodied with a portfolio of methods and provided intelligence to select the best processing technique dynamically according to the context's density. The experimental results indicate that by implementing simple behaviors in methods, the system can, without prior knowledge of image processing, offer vehicle tracking quality comparable to the best image analysis approaches. This is achieved with significant gains in runtime performance and minimal agent interaction costs. Future work will involve introducing an organization responsible for the system's learning for continual learning of methods.

ACKNOWLEDGEMENTS

Dani Manjah is supported by the Walloon region under grant n°1910045 – APTITUDE by Win2WAL. This work was also supported by Wallonia-Brussels International, the Fund for Scientific Research, the French Ministry of Foreign and European Affairs, and the Ministry of Higher Education and Research within the framework of the Hubert Curien Partnerships.

REFERENCES

Abbas, H. A. (2015). Organization of multi-agent systems: An overview. *International Journal of Intelligent Information Systems*, 4(3):46.

Abdelrahim, M., Hendrickx, J. M., and Heemels, W. P. (2018). MAX-consensus in open multi-agent systems with gossip interactions. *2017 IEEE 56th Annual Conference on Decision and Control, CDC 2017*, 2018-January:4753–4758.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468.

Bochkovskiy, A., Wang, C., and Liao, H. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934.

Chen, C.-C., Hsu, J.-M., and Liao, C.-H. (2009). Hama: A three-layered architecture for integrating object tracking and location management in wireless sensor networks. In *2009 Third International Conference on Multimedia and Ubiquitous Engineering*, pages 268–275.

Cossentino, M., Gaud, N., Hilaire, V., Galland, S., and Koukam, A. (2010). Aspecs: an agent-oriented software process for engineering complex systems. *Autonomous Agents and Multi-Agent Systems*, 20(2):260–304.

Elharrouss, O., Almaadeed, N., and Al-Maadeed, S. (2021). A review of video surveillance systems. *Journal of Visual Communication and Image Representation*, 77:103116.

Feraud, M. and Galland, S. (2017). First comparison of sarl to other agent-programming languages and frameworks. *Procedia Computer Science*, 109:1080 – 1085. 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal.

Ferber, J., Gutknecht, O., and Michel, F. (2003). From agents to organizations: an organizational view of multi-agent systems. In *Agent-Oriented Software Engineering IV*, volume 2935, pages 214–230, Berlin, Heidelberg. Springer Berlin Heidelberg.

GALLAND, S., MUALLA, Y., TCHAPPI HAMAN, I., ZHAO, H., RODRIGUEZ, S., NAJJAR, A., and GAUD, N. (2019). Model transformations from the sarl agent-oriented programming language to an object-oriented programming language. *International Journal on Agent-Oriented Software Engineering*, 1–38. Rank: Q2.

Gerber, C., Siekmann, J., and Vierke, G. (1999). Flexible autonomy in holonic multiagent systems. In *AAAI Spring Symposium on Agents with Adjustable Autonomy*.

Hendrickx, J. M. and Martin, S. (2018). Open multi-agent systems: Gossiping with random arrivals and departures. *2017 IEEE 56th Annual Conference on Decision and Control, CDC 2017*, 2018-January:763–768.

Hilal, A. R. and Basir, O. A. (2015). A scalable sensor management architecture using BDI model for pervasive surveillance. *IEEE Systems Journal*, 9(2):529–541.

Liu, Y.-C., Tian, J., Glaser, N., and Kira, Z. (2020). When2com: Multi-agent perception via communication graph grouping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Luiten, J., Osep, A., Dendorfer, P., Torr, P., Geiger, A., Leal-Taixé, L., and Leibe, B. (2021). Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, 129:1–31.

Luo, Z., Branchaud-Charron, F., Lemaire, C., Konrad, J., Li, S., Mishra, A., Achkar, A., Eichel, J., and Jodoin, P.-M. (2018). Mio-tcd: A new benchmark dataset for vehicle classification and localization. *IEEE Transactions on Image Processing*, 27(10):5129–5141.

Mihailescu, R. C., Davidsson, P., Eklund, U., and Persson, J. A. (2018). A survey and taxonomy on intelligent surveillance from a system perspective. *The Knowledge Engineering Review*, 33(1).

Naphade, M., Wang, S., Anastasiu, D. C., Tang, Z., Chang, M.-C., Yang, X., Yao, Y., Zheng, L., Chakraborty, P., Lopez, C. E., Sharma, A., Feng, Q., Ablavsky, V., and Sclaroff, S. (2021). The 5th ai city challenge. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Park, H. D., Min, O. G., and Lee, Y. J. (2017). Scalable architecture for an automated surveillance system using edge computing. *Journal of Supercomputing*, 73(3):926–939.

Perera, C., Zaslavsky, A., Christen, P., and Georgakopoulos, D. (2014). Sensing as a service model for smart cities supported by Internet of Things. *Transactions on Emerging Telecommunications Technologies*, 25(1):81–93.

Rao, B., Durrant-Whyte, H. F., and Sheen, J. (1993). A fully decentralized multi-sensor system for tracking and surveillance. *The International Journal of Robotics Research*, 12(1):20–44.

Rodriguez, S., Hilaire, V., Gaud, N., Galland, S., and Koukam, A. (2011). Holonic Multi-Agent Systems. *Natural Computing Series*, 37:251–279.

Rombach, H. D., Basili, V. R., and Selby, R. W. (1993). *Experimental Software Engineering Issues:: Critical Assessment and Future Directions. International Workshop, Dagstuhl Castle, Germany, September 14-18, 1992. Proceedings*, volume 706. Springer Science & Business Media.

Saffre, F. and Hildmann, H. (2021). Adaptive behaviour for a self-organising video surveillance system using a genetic algorithm. *Algorithms*, 14(3):1–11.

Simon, H. A. (1976). How complex are complex systems? *PSA: Proceedings of the Biennial Meeting of the Philosophy of Science Association*, 1976(2):507–522.

Soviany, P., Ionescu, R. T., Rota, P., and Sebe, N. (2021). Curriculum self-paced learning for cross-domain object detection. *Computer Vision and Image Understanding*, 204:103166.

Valencia-Jiménez, J. J. and Fernández-Caballero, A. (2006). Holonic multi-agent systems to integrate independent multi-sensor platforms in complex surveillance. *Proceedings - IEEE International Conference on Video and Signal Based Surveillance 2006, AVSS 2006*, pages 49–54.

Wautelet, Y., Schinckus, C., and Kolp, M. (2021). Agent-based software engineering, paradigm shift, or research program evolution. In *Research Anthology on Recent Trends, Tools, and Implications of Computer Programming*, pages 1642–1654. IGI Global.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2):115–152.

APPENDIX

```
Data: CPP (CyberPhysicalPlatform)
Result: Managed interactions within CPP
Holarch CPP:
if receive external request for perceptions then
    ResourceProvider.manageRequest();
    Observer.initiateProcessing();
end
Role ResourceProvider:
if new request or mission then
    if resources available and legitimate request then
        allocate resources;
        authorize interactions;
    else
        deny request;
    end
end
Role Observer:
if initiation then
    while mission is active do
        Sensor.acquireData();
        process data to generate perceptions;
        if collaboration or competition is needed then
            interact with other Observers;
        end
    end
end
Role Sensor:
if requested by Observer then
    if data relies on another CPP then
        request data from other CPP;
    end
    Provide data to Observer;
end
```

Algorithm 1: Holonic Organizational Interactions in CPP.

```
Data: costX, perfX, densityInfo
Result: Adaptive behavior based on density, cost, and performance
Agent Algorithm - Method X:
Behavior:
if low density and costX is min then
    Act as primary detector;
else
    if average density and costX is min then
        Act as activity detector;
        if activity detected then
            Notify other method;
        end
    else
        if high density and perfX is max then
            Act as primary detector;
        end
    end
end
Interaction & Decision Protocol:
Share cost and performance;
Evaluate current density;
Apply rules based on density, cost, and performance;
```

Algorithm 2: Rule-based Interactions in Implemented CPP for the level n-2.