INNOVATION BASED TRANSMISSION IN AI-ENABLED SENSOR NETWORKS FOR 6G IOT SCENARIO

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

By

Ahmet Arda Atalık

September 2022

Innovation based transmission in AI-enabled sensor networks for 6G IoT scenario
By Ahmet Arda Atalık
September 2022

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Orhan Arıkan(Advisor)

Tolga Mete Duman

Serkan Sarıtaş

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

INNOVATION BASED TRANSMISSION IN AI-ENABLED SENSOR NETWORKS FOR 6G IOT SCENARIO

Ahmet Arda Atalık
M.S. in Electrical and Electronics Engineering
Advisor: Orhan Arıkan
September 2022

Goal-oriented signal processing and communications are assured to play a key role in developing the next generation of sensor devices and networks, e.g., 6G IoT networks. A critical task of semantic signal processing is the detection of innovation and transmission scheduling based on the innovation in AI-enabled sensor networks and IoT for 6G. This thesis proposes efficient and optimal sampling and transmission strategies for goal-oriented sensor networks for various data models, investigates their performances both analytically and numerically; and introduces the use of dimensionality reduction algorithms in semantic signal processing and highlights its effectiveness in real life case studies. That is, the proposed methods are explained rigorously and demonstrated through simulations and case studies based on real-world computer vision examples with recorded video signals. Numerical results indicate that the next generation sensor devices and networks can benefit significantly from the proposed methods in terms of energy efficiency and semantic innovation detection performances.

Keywords: goal-oriented signal processing, optimal sampling.

ÖZET

YAPAY ZEKA ETKİNLENMİŞ SENSÖR AĞLARINDA 6G NESNELERİN İNTERNETİ SENARYOLARI İÇİN İNOVASYON BAZLI İLETİM

Ahmet Arda Atalık
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Orhan Arıkan
Eylül 2022

Hedefe yönelik sinyal işleme ve iletişim gelecek nesil (6G) sensör ağ ve aygıtlarının geliştirilmesinde anahtar bir rol oynayacaktır. Hedefe yönelik sinyal işlemede kritik bir iş paketi ise inovasyon kestirimi ve inovasyon bazlı iletim yöntemlerinin akıllı sensör ve nesnelerin interneti ağlarında 6G senaryoları için kullanımıdır. Bu tez hedefe yönelik sensör ağları için verimli ve eniyi örnekleme ve iletim stratejileri ileri sürüp bunların performanslarını analitik ve nümerik olarak inceler ve anlamsal sinyal işlemede boyut indirgeme algoritmalarının kullanımını öne sürerek gerçek hayat uygulamalarında bu metotların etkinliğini vurgular. Önerilen metotlar dikkatli bir biçimde açıklandıktan sonra gerçek hayat bilgisayar görüşü örnekleri ve vaka analizleriyle desteklenir. Yapılan nümerik çalışmalar gelecek nesil sensör ağ ve aygıtlarının ileri sürülen metotlardan enerji verimi ve anlamsal inovasyon kestirimi bakımından belirgin biçimde yararlanabileceğini ortaya koymaktadır.

Anahtar sözcükler: hedef odaklı sinyal işleme, eniyi örnekleme.

Acknowledgement

First, I would like to thank Prof. Orhan Arıkan for his considerable support and supervision throughout my M.S. studies. I would like to convey my sincere gratitude to the examination committee members Prof. Tolga Mete Duman and Prof. Serkan Sarıtaş.

From Bilkent University, I thank Prof. Gökhan Yıldırım, Prof. Sinan Gezici, Prof. Erdal Arıkan, and all my professors for their contributions to my undergraduate and graduate years through their courses and research supervisions.

I would like to thank the members of our research group; Prof. Tolga Mete Duman, Mert Kalfa, Mehmetcan Gök, Sadık Yağız Yetim and Büşra Tegin for their support and collaboration.

As a tradition, I express my deepest regards to Doğancan Çiçek, Emir Ceyani, and Mahmut Ercan for their firm friendships.

Last but not least, I want to thank and express my deepest gratitude to my family and my girlfriend Aslı for their continuous support and encouragement throughout my studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The *Internet of Things (IoT)* is a wireless communication paradigm that has gained significant attention in the last decade. The main idea behind is the ubiquitous existence of a collection of *things* or *objects*, e.g., smart sensors, actuators, alarms, smartwatches, smart home appliances, and radio-frequency identification (RFID) tags, which are capable of communicating with each other to attain common goals [4, 5]. Potential application areas of IoT can be analyzed under four main domains, i.e., transportation and logistics, healthcare, smart environment, and social domains [5, 6]. In logistics, RFID and NFC information flow allows real-time monitoring in each supply-chain link. In transportation, smart vehicles and smart roads/rails equipped with various sensor networks provide enhanced and reliable navigation and safety. Similarly, tracking and sensing capabilities of RFID also make IoT suitable for healthcare and smart environment applications.

Two foundational discoveries for IoT are the RFID technology and wireless sensor networks (WSNs): RFID has been used in logistics, production, and supply chain since 1980s and WSNs have been used in environmental, industrial, and healthcare monitoring since 1990s [6]. Fig. 1.1 illustrates the evolution of the IoT. Other enabling technologies for IoT are smartphones, barcodes, near-field communication, WiFi, internet IPv6, social networks, 3G, 4G, and most importantly 5G and beyond [1].

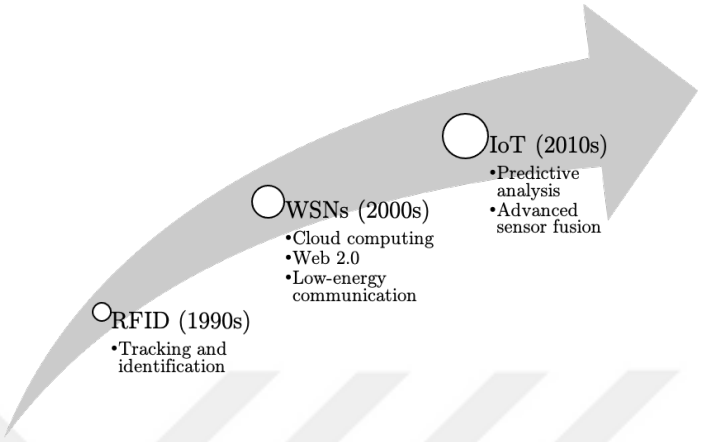


Figure 1.1: Evolution of the IoT [1]. Inside the parentheses are the beginning decades.

3G and 4G networks have been used extensively in the IoT and 5G and beyond networks are expected to expand the IoT applications massively [7, 8]. As per the applications requirements, 5G and beyond networks emphasize new performance criteria, e.g., ultra-low latency, throughput, power, and massive connectivity. According to reports, it is estimated that 70% of companies spent over $1.2 billion for connectivity and 6.1 and 11.3 billion IoT devices was connected in 2017 and 2020, respectively [7, 8, 2]. The same report forecasts the number of connected IoT devices by 2025 as at least 27 billion. Fig. 1.2 illustrates the number of connections in different years.

As the number of connected IoT devices grows rapidly, the efficiency becomes an important research area and its importance is overemphasized in various surveys [7, 9]. In this thesis, the energy efficiency is investigated for goal-oriented IoT sensor networks and semantic signal processing and communication applications. The analyses carried on show that certain tasks can be implemented in a goal-oriented setting more efficiently than using off-the-shelf methods, and semantic processing can benefit significantly from the existing dimensionality reduction schemes.

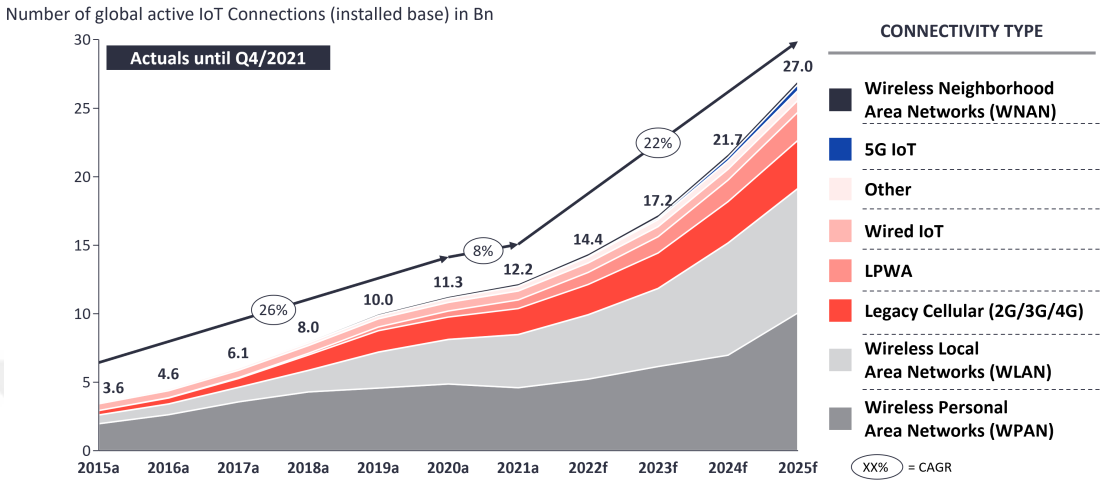


Figure 1.2: Global IoT market forecast [2]. On the x-axis, "a" stands for actual, and "f" stands for forecast.

1.1 Organization of the Thesis

Chapter 2 develops efficient and optimal sampling and transmission strategies for goal-oriented sensor networks for various data models and investigates their performances both analytically and numerically, and compares their performance with the standard sampling methods.

Chapter 3 introduces the use of dimensionality reduction algorithms in semantic signal processing and communication, and provides various case studies to highlight its effectiveness in massive IoT settings.

Chapter 4 concludes the thesis and suggests some further research directions.

1.2 Notation

Throughout the thesis, lowercase and uppercase bold letters, i.e., $\mathbf{x}, \mathbf{X}$ are used for vectors and matrices, respectively. Time indices for discrete-time are denoted with a subscript n, e.g., x_n, or $x[n]$ and the $p-$norm of a vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_p$. Probability of an event and expectation of a random variable are denoted

by double struck initials, e.g., $\mathbb{P}(\cdot)$, $\mathbb{E}[\cdot]$, respectively.

Chapter 2

Efficient and Optimal Sampling Strategies for Distributed Sensors

2.1 Introduction

Energy efficiency is a critical research area for sensors, especially for the next generation of communication networks where the number of IoT devices is expected to rapidly increase. The proposed goal-oriented semantic language and the signal processing framework can enable dramatic reductions in transmitted information, especially when the sampled phenomenon is changing slowly with respect to the goals defined either internally or externally, leading to significant energy savings.

Another avenue of improvement for the efficiency of sensors can be introduced by employing smart non-uniform sampling strategies. For scalar sensors, if the signal output can be modeled as a random process whose statistical characterization is available, the sampling times can be adjusted according to desired *detection* and *missed detection* probabilities of certain events, e.g., a threshold crossing.

The aim is to choose the sampling times to trade-off between energy consumption and misdetection of the exceedings. The number of samples must be kept minimum to reduce energy consumption, and the probability of missing the threshold must be less than or equal to a given error.

Rest of this chapter is organized as follows: We present a novel sampling strategy for a generic scalar sensor output, in conjunction with a goal-oriented semantic representation to improve the energy efficiency of sampling and transmission operations. First, we introduce a novel and optimal sampling strategy for scalar sensor outputs that conform to a simple symmetric random walk and then we proceed to a Gaussian random walk and a constant-variance auto-regressive AR(p) Gaussian model in which the goal is defined as tracking threshold crossings. We present statistical results on the improved performance.

2.2 Preliminaries and Data Models

We have a discrete time signal $x[n]$ over the horizon H, i.e., $n = 0 : H-1$ where $x[n]$ comes from a generic p^{th} order auto-regressive (AR(p)) process. That is,

$$x[n] = \sum_{i=1}^{p} \phi_i \, x[n-i] \; + \; w[n], \tag{2.1}$$

where p and ϕ_i is fixed and $w[n]$ is independent of $x[n-1], x[n-2], \ldots, x[n-p]$ for every n. Our aim is to develop sampling strategies for single or double threshold crossings so that the misdetection of the exceeding is smaller than a tolerable level for various assumptions on $w[n]$, p, and ϕ_i.

2.3 Optimal Sampling Strategies

We begin our analysis by a simple random walk with single threshold, and move towards more practical and complicated models such as constant-variance AR(p) with a lower and upper thresholds.

2.3.1 Simple Random Walk

In this subsection, we assume the observed data follows a simple symmetric random walk model:

$$x[n] = x[n-1] + w[n], \tag{2.2}$$

where $w[n] = 2b[n] - 1$, $b[n] \sim \text{Bern}(0.5)$ and without loss of generality $x[0] = 0$. Suppose we have a single threshold r to be detected, i.e., we would like to raise an alarm if a sample hits or exceeds a threshold $r > 0$. Initially, the gap to the threshold is $g = r$. Our aim is to choose sampling times such that the probability of missing the threshold crossing is less than or equal to a given miss probability. Obviously, no missing occurs if we take the first sample at a time on or before r. If we take it after, a case-by-case analysis is necessary.

- Taking the first sample at $r+1$ or $r+2$ misses the threshold if $x[r] = r$, which happens with probability 2^{-r}.

- Taking the first sample at $r+3$ or $r+4$ misses if $x[r] = r$ or $x[r+2] = r$. Let us calculate its probability:

$$\begin{aligned}
\mathbb{P}\left(\{x[r] = r\} \cup \{x[r+2] = r\}\right) &= \mathbb{P}\left(\{x[r] = r\}\right) + \mathbb{P}\left(\{x[r+2] = r\}\right) \\
&\quad - \mathbb{P}\left(\{x[r] = r\} \cap \{x[r+2] = r\}\right) \\
&= 2^{-r} + \binom{r+2}{1} 2^{-(r+2)} - 2^{-(r+1)} \\
&= 2^{-r}\left(1 + \frac{r}{4}\right).
\end{aligned}$$

- Taking the first sample at $r+5$ or $r+6$ misses if $x[r] = r$, or $x[r+2] = r$, or $x[r+4] = r$. We can calculate its probability using the inclusion-exclusion principle, the result would be $2^{-r}\left(1 + \frac{11}{32}r + \frac{1}{32}r^2\right)$.

- Taking the first sample at $r+7$ or $r+8$ misses with probability $2^{-r}\left(1 + \frac{152}{384}r + \frac{21}{384}r^2 + \frac{1}{384}r^3\right)$.

- For any positive even integer k, sampling at $r+k-1$ or $r+k$ introduces an error $2^{-r}\, P_{\frac{k}{2}-1}$, where P_n is an n^{th} order polynomial.

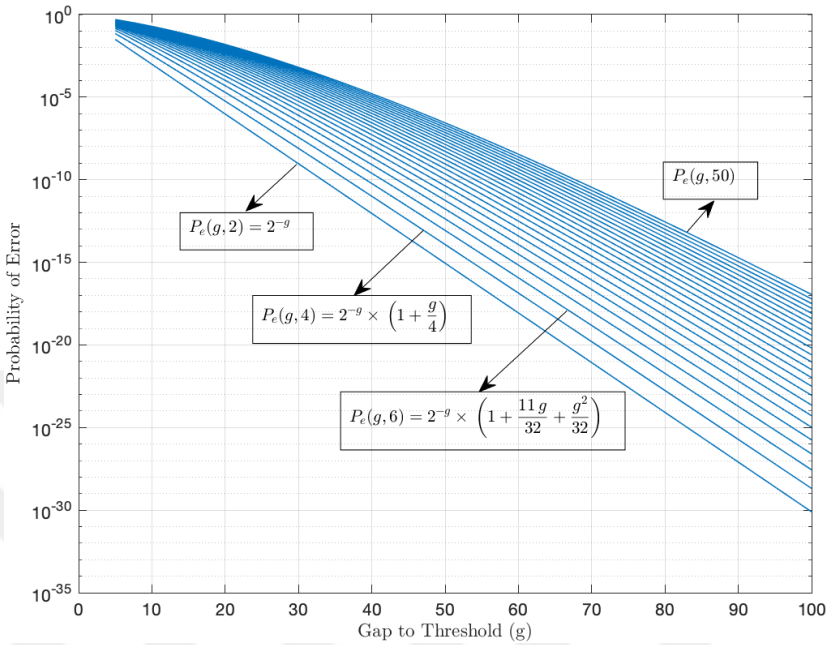
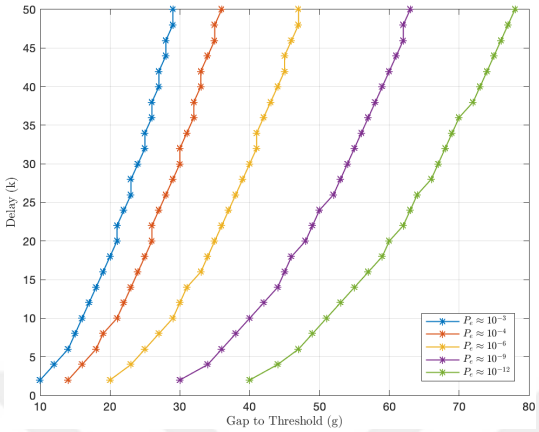


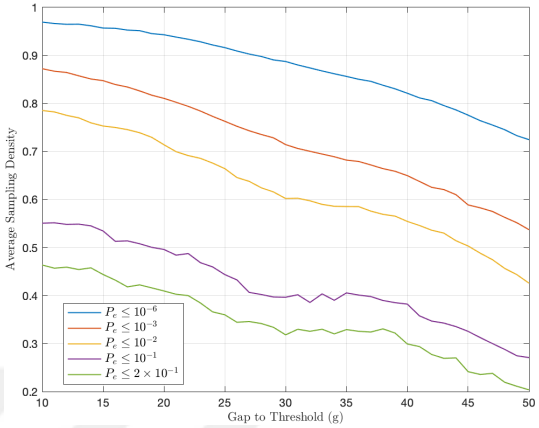
Figure 2.1: Probability of error vs. threshold for various sampling delays for simple random walk

The coefficients of $P_{\frac{k}{2}-1}$ for any delay k can be calculated recursively. As sampling at $r+k-1$ or $r+k$ yields the same error for any positive integer k, we sample at $r+k$. Fig. 2.1 illustrates the relation between the sampling delay and the probability of missing the exceeding.

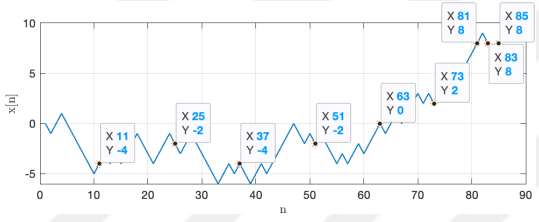
Another important plot is the delay versus gap to the threshold for various miss probabilities, illustrated in Fig. 2.2(a). Using this data, it is easy to execute the optimal sampling strategy for different miss probabilities. Assume the previous sample has taken at n_l. To find the next sampling time n_{l+1}, we update the gap to threshold g with $\left|r - x[n_l]\right|$ and find the corresponding delay for a tolerable miss probability. In Fig. 2.2(c), we implement the error-free strategy and the 0.001 miss probability strategy for a realization of a simple symmetric random walk where the threshold is set to 10. Over a 90 unit time interval, zero-error and 0.001 miss probability strategy takes 9 and 7 samples, respectively. Obviously, as the random walk moves closer to the threshold, the sampling frequency increases, and vice versa. Averaging the number of samples over a thousand of random walk realizations, we obtain the Fig. 2.2(b).

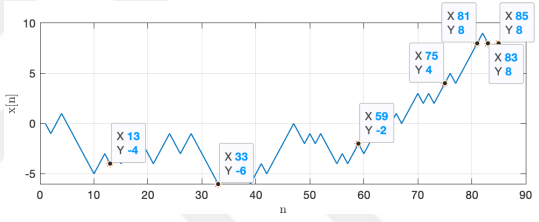


(a) Delay vs. gap to threshold for various probability of errors



(b) Average sampling density vs. gap to threshold





(c) Optimal sampling strategy for zero error and 0.001 miss probability for simple random walk

Figure 2.2: Delay and average sampling density vs. gap to threshold and optimal sampling strategy illustration for simple random walk

2.3.2 Gaussian Random Walk

In this subsection, we assume that $w[n]$ in (2.2) is a standard Gaussian random variable. That is, $x[n] = x[n-1] + w[n-1]$ for $n \geq 1$ and $w[i]$ are i.i.d. standard normal random variables. We use the same notation: the threshold is denoted with r, and the gap to the threshold at time n is denoted with $g[n] = |r - x[n]|$.

Define $\tau_r = \inf\{n > 0 : x[n] > r\}$ as the first exceeding time. Assume we take the first sample at time $n_1 = 1 + k$ for some non-negative k, and define its misdetection probability as

$$P_{md}(k, r) = \mathbb{P}(\tau_r \leq k). \tag{2.3}$$

Calculation of $P_{md}(k, r)$ as a function of r for fixed ks can be done by observing $\{\tau_r = k\}$ if and only if $\{x[k] > r, x[l] \leq r \quad \forall l < k\}$. For every $n > 1$, define

$h_n(r) = \mathbb{P}(x[n] \leq r \,\big|\, x[n-1] \leq r)$. By the Law of total probability and Markov property:

$$\mathbb{P}(\tau_r = k) = \begin{cases} (1 - h_k(r))\ \Phi(r)\ \prod_{i=2}^{k-1} h_k(r) & k > 1 \\ 1 - \Phi(r) & k = 1 \end{cases},$$

$$P_{md}(k, r) = 1 - \Phi(r) \prod_{n=2}^{k} h_n(r) \quad \forall k \geq 1,$$

where $h_n(r) = \dfrac{\int_{-\infty}^{\frac{r}{\sqrt{n-1}}} \Phi\left(r - \sqrt{n-1}\, z\right)\ f(z)\, dz}{\Phi\left(\frac{r}{\sqrt{n-1}}\right)}$ $\forall n > 1$, and $f(z)$ is the PDF of a standard normal random variable (rv), i.e., $f(z) = \frac{1}{\sqrt{2\,\pi}} \exp(-z^2/2)$. Similar to the symmetric random walk, we can obtain the misdetection vs. gap to the threshold for various delays, and delay vs. gap to the threshold for various miss probabilities, and average sampling density vs. gap to the threshold as in Fig. 2.3. Using these, we implement the sampling strategy and illustrate it in Fig. 2.4.

2.3.3 Constant Variance AR(p) Models

2.3.3.1 Gaussian Innovations

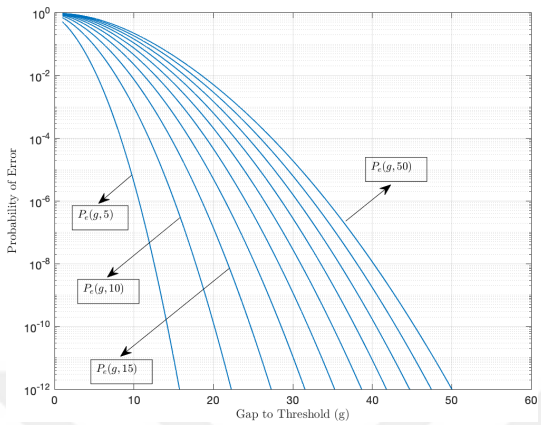
As the variance of $x[n]$ in the Gaussian random walk is increasing linearly with n, the model becomes more unrealistic as the time increases. One solution to this problem is to choose ϕ_i in (2.1) so that $\mathrm{Var}(x[n])$ is a constant independent of n. For a first order process, this can be accomplished with

$$x[n+1] = \alpha\, x[n] + \sqrt{1 - \alpha^2}\, w[n] \text{ for } n \geq 0, \tag{2.4}$$

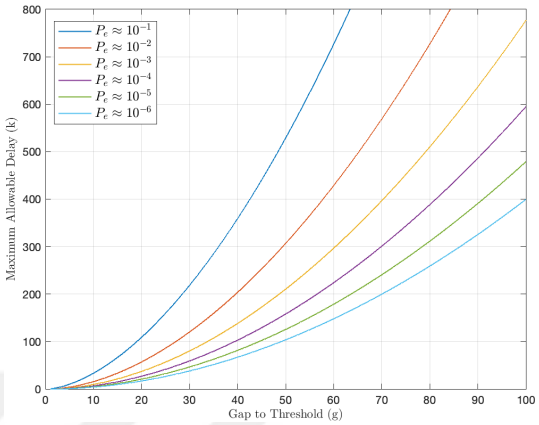
where $\alpha \in (0, 1)$, $x[0] \sim \mathcal{N}(0, 1)$ and $w[i] \sim \mathcal{N}(0, 1)$. For a p^{th} order process, the model becomes

$$x[n+1] = \sum_{k=1}^{p} \alpha_k\, x[n+1-k] + \sqrt{1 - \sum_{k=1}^{p} \alpha_k^2}\, w[n], \tag{2.5}$$

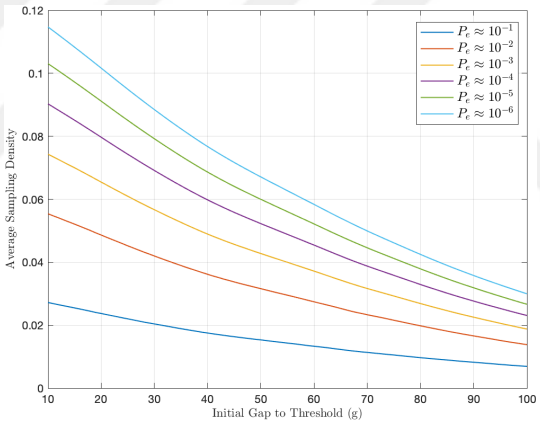
with proper initial conditions such that $\mathbb{E}[x_i^2] = \mathbb{E}[x_j^2]$ for all i, j.



(a) Probability of error vs. gap to threshold for various delays



(b) Delay vs. gap to threshold for various probability of errors



(c) Average sampling density vs. gap to threshold

Figure 2.3: Probability of error, delay and average sampling density vs. gap to threshold for Gaussian random walk

2.3.3.1.1 The Case of Single Threshold Crossing

For the $p^{th}-$order Gaussian AR process in (2.5), let $\boldsymbol{X} \triangleq [X_1, X_2, \ldots, X_N]^T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_N)$. Define

$$\Phi_N(\boldsymbol{u}, \boldsymbol{\Sigma}_N) = \mathbb{P}\left(X_1 \leq u_1, X_2 \leq u_2, \ldots, X_N \leq u_N\right),$$

where $\Phi_1(u, \sigma^2) = \Phi\left(\frac{u}{\sigma}\right)$, and

$$h_p(r) = \frac{\Phi_p(r\,\mathbf{1}_p, \boldsymbol{\Sigma}_p)}{\Phi_{p+1}(r\,\mathbf{1}_{p+1}, \boldsymbol{\Sigma}_{p+1})}.$$

Assume we take the first sample at time $n_1 = 1 + k$ for some non-negative k, the misdetection probability is $P_{md}(k, r) = \mathbb{P}(\tau_r \leq k)$ where we use the same

definition of the first exceeding time τ_r. It is easy to see the relation between the CDF of τ_r and $h_p(r)$:

$$\mathbb{P}\left(\tau_r \leq n\right) = 1 - \left(h_p(r)\right)^{-n}. \tag{2.6}$$

We are now ready to the state the optimal sampling time theorem.

Theorem 1 (Optimal sampling time in the case of single threshold crossing)**.** *Given the tolerable misdetection probability p_e, the optimal inter-sampling time is calculated as*

$$n^*(r) \triangleq \sup\left\{n \geq 1 \,\middle|\, \mathbb{P}(\tau_r \leq n-1) \leq p_e\right\} \tag{2.7}$$

$$= \left\lceil \frac{\log\left(\frac{1}{1-p_e}\right)}{\log(h_p(r))} \right\rceil. \tag{2.8}$$

Hence, for an initial sample x_0 below the threshold r, the initial and the $(k+1)^{th}$ sampling time is as follows.

$$n_1 = \left\lceil \frac{\log\left(\frac{1}{1-p_e}\right)}{\log(h_p(|x_0 - r|))} \right\rceil, \tag{2.9}$$

$$n_{k+1} = n_k + \left\lceil \frac{\log\left(\frac{1}{1-p_e}\right)}{\log(h_p(|x_{n_k} - r|))} \right\rceil \quad \forall k > 0. \tag{2.10}$$

Proof. Follows directly from the equation (2.6) and the definition of optimal sampling time (2.7). □

For the calculation of $h_p(\cdot)$, multivariate normal CDF packages exist in many programming languages, e.g., `mvncdf` in MATLAB and `mvtnorm` in R. Moreover, in bivariate and trivariate cases, several closed form approximations exist. For $p > 3$, quasi-Monte Carlo integration algorithms are utilized. For $p = 1$, an approximation is

$$h(r) \approx \frac{\Phi(r)}{\Phi(r) - \frac{1}{2\,\pi}\,\arccos\left(\alpha\right)\exp\left(-\frac{r^2}{2}\right)}. \tag{2.11}$$

We illustrate the gap to threshold vs. sampling time in a constant-variance AR(1) Gaussian process for various misdetection probabilities in Fig. 2.5. Next, we discuss some statistical properties of the sampling algorithm in Lemma 1 and Corollary 2, whose proofs are provided below.

Lemma 1 (Heavy tails of inter-sampling time)**.**
For constant-variance Gaussian AR(p) processes, given any threshold r, miss-probability p_e, and correlation coefficients $\alpha_1, \alpha_2, \ldots, \alpha_p$:

$$\mathbb{P}(n_1 > k) \sim \Omega\left(\frac{1}{k}\right) \quad \text{as } k \to \infty \implies \mathbb{E}(n_1) = \infty. \tag{2.12}$$

Proof.

$$\mathbb{P}\left(n_1 > k\right) = \mathbb{P}\left(h_p(|x_0 - r|) < \left(\frac{1}{1-p_e}\right)^{\frac{1}{k}}\right) \tag{2.13}$$

$$= \Phi\left(r - h_p^{-1}\left(\left(\frac{1}{1-p_e}\right)^{\frac{1}{k}}\right)\right) + \Phi\left(-r - h_p^{-1}\left(\left(\frac{1}{1-p_e}\right)^{\frac{1}{k}}\right)\right) \tag{2.14}$$

$$\simeq \frac{\exp\left(-\frac{1}{2}\left(r - h_p^{-1}\left(\left(\frac{1}{1-p_e}\right)^{\frac{1}{k}}\right)\right)^2\right)}{h_p^{-1}\left(\left(\frac{1}{1-p_e}\right)^{\frac{1}{k}}\right)\sqrt{2\pi}}. \tag{2.15}$$

Then, the result follows from the Taylor series expansion of $h_p(r)$. □

Corollary 2 (Sublinear growth of number of samples)**.**
For a finite horizon $H > 0$, Define $N_s(H) = \sup\{k \geq 0 : n_k \leq H\}$ and $m_H = \sum_{k=0}^{H} \mathbb{P}(n_1 > k)$. Elementary Renewal Theorem [10] guarantees

$$\mathbb{E}(n_1) = \infty \implies \frac{\mathbb{E}(N_s(H))}{H} \searrow 0. \tag{2.16}$$

Hence, asymptotically, the expected number of samples grows sub-linearly. Since $k\,\mathbb{P}(n_1 > k)$ is a slowly varying function by Lemma 1, Strong Renewal Theorems [11] gives:

$$\frac{\mathbb{E}(N_s(H))}{H} \sim \frac{1}{m_H}. \tag{2.17}$$

Moreover, $m_H \gg \log H$. So, the sampling density decays faster than $\dfrac{1}{\log H}$.

2.3.3.1.2 The Case of Double Threshold Crossing

For a double threshold event detection problem, we define $\tau \triangleq \inf\left\{n > 0 :\ x_n \notin (l, u)\right\}$ where u, l are the upper and lower thresholds, respectively. Assume that we take the first sample at time $n_1 = 1 + k$ for some non-negative k, and denote its probability of missing a threshold crossing as $P_{md}(k, l, u)$. We can relate this error probability with the CDF of τ as

$$\begin{aligned} P_{md}(k, l, u) &= \mathbb{P}(\tau \leq k) \\ &= 1 - \mathbb{P}\left(\left\{\sup_{s \leq k} x_s < u\right\} \bigcap \left\{\inf_{s \leq k} x_s > l\right\}\right) \\ &= 1 - \mathbb{P}\left(x_s \in (l, u) \quad \forall s \in \{1, 2, \ldots, k\}\right) \\ &= 1 - \psi_k, \end{aligned} \tag{2.18}$$

where $\psi_k \triangleq \mathbb{P}\left(x_s \in (l, u) \quad \forall s \in \{1, 2, \ldots, k\}\right)$ with $\psi_0 = 1$ as a convention.

Given a tolerable missed detection probability p_m, we choose the largest $k \in \mathbb{N}$ such that $\psi_k \geq 1 - p_m$ and take the first sample at $n_1 = 1 + k$. Since x_n follows an AR(p) process, calculation of ψ_k can be done by using the Markov Property [12], i.e., factorization of the joint probability using conditional independence.

Next, we illustrate the derivation of the sampling algorithm for an AR(1) model. Generalization to a p-th order AR process can be performed similarly. We consider the model in (2.4) assuming that $x_0 \in (l, u)$ is observed. Therefore, we have $\mathbb{E}\left[x_n \mid x_0\right] = \alpha^n x_0$, $\mathbb{V}\left[x_n \mid x_0\right] = 1 - \alpha^{2n}$, and $\mathbb{C}\left[x_n,\, x_{n-1} \mid x_0\right] = \alpha\left(1 - \alpha^{2(n-1)}\right)$, i.e., the mean is strictly decreasing towards 0 while the variance and covariance are strictly increasing towards 1 and α, respectively.

To be able to calculate the next optimal sampling instance, we define the probability of the i-th sample being inside the double threshold given that the previous sample was also inside the thresholds as

$$h_i \triangleq \mathbb{P}\left(x_i \in (l, u) \,|\, x_{i-1} \in (l, u)\right), \tag{2.19}$$

with an initial condition of $h_1 = \mathbb{P}\left(x_1 \in (l, u)\right)$. Calculation of (2.19) requires only the first and second order statistics of x_{i-1} and x_i, which can be computed

numerically. Finally, the next optimal sampling time can be calculated as

$$n_1 = \inf \left\{ n \in \mathbb{Z}_+ \,\middle|\, \psi_n < 1 - p_m \right\}, \tag{2.20}$$

where ψ_k is defined using the Markov Property as

$$\psi_k = \prod_{i=1}^{k} h_i. \tag{2.21}$$

Once we observe the process at time n_1, we proceed to calculate the next sampling time n_2, and for any $k \in \mathbb{Z}^+$, the algorithm uses x_{n_k} to determine the next optimal sampling time n_{k+1}.

In Fig. 2.6, we illustrate the proposed sampling algorithm for a first-order Gaussian AR process. Our aim is to detect crossings of thresholds $u = 2$ and $l = -2$. The desired maximum probability of missing the threshold crossings is set at 0.05. As expected, when the minimum distance to thresholds ($\min\{|x_n - l|, |u - x_n|\}$) decreases, the sampling rate increases.

We now investigate various statistical properties of (2.20) under the assumption

$$x_0 \sim \mathcal{N}(0,1) \text{ is within the boundaries, i.e., } x_0 \in (l,u). \tag{2.22}$$

The distribution of the optimal sampling times requires calculation of (2.21), i.e., the products of dependent random variables h_i, under the assumption given in (2.22). The expectation of n_1 has significance since it represents the *average sampling period* of the algorithm. The expressions for the CDF and the expectation of n_1 are as follows:

$$\mathbb{P}\left(n_1 \leq k \,|\, x_0 \in (l,u)\right) = \mathbb{P}\left(\prod_{i=1}^{k} h_i < 1 - p_m\right), \tag{2.23}$$

$$\mathbb{E}\left[n_1 \,|\, x_0 \in (l,u)\right] = 1 + \sum_{k=1}^{\infty} \mathbb{P}\left(\prod_{i=1}^{k} h_i \geq 1 - p_m\right), \tag{2.24}$$

which can be computed numerically for given threshold levels l, u and missed detection probability p_m.

Next, we present two simulation results. In the first simulation, we investigate the expected sampling period for various missed detection probabilities for AR(1) models. We use a first-order Gaussian AR process as in (2.4) with $\alpha = 0.95$, i.e.,

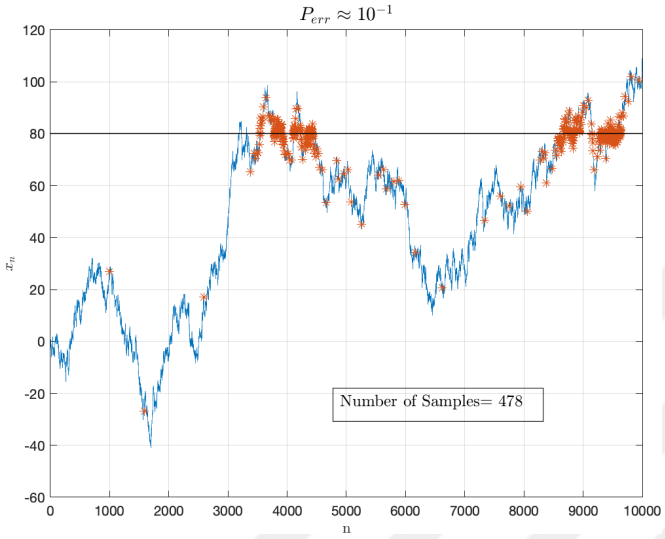
$$x_n = 0.95\, x_{n-1} + \sqrt{1 - 0.95^2}\, w_{n-1} \text{ for } n \geq 1, \qquad (2.25)$$
$$x_0 \sim \mathcal{N}(0,1) \text{ and } w_i \sim \mathcal{N}(0,1).$$

Setting the thresholds $l = -2$, $u = 2$, we plot the expected first sampling time $\mathbb{E}\left[n_1 \,|\, x_0 \in (l,u)\right]$ versus the tolerated maximum probability of missed detections under the assumption that the current sample is within the boundaries in Fig. 2.7(a). As expected, the sampling period is positively correlated with the missed detection probability. The relationship is observed to be almost linear for this example.

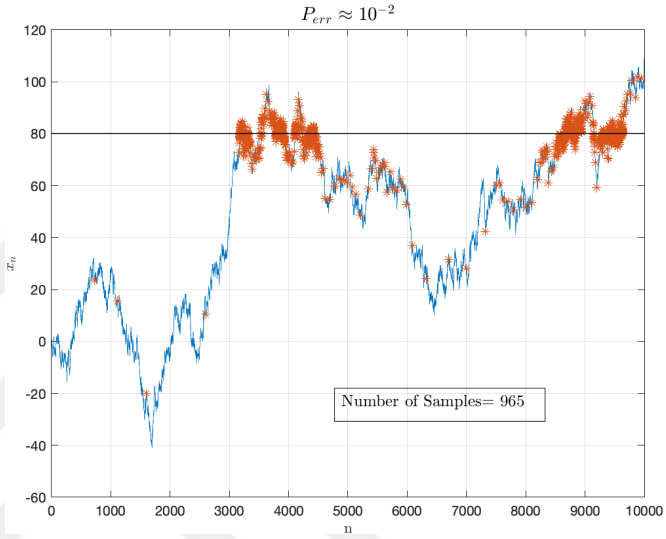
In Fig. 2.7(b), we present the effect of the current sample value on the next sampling time interval for different missed detection probabilities p_m. The underlying assumptions and the model are the same as those in the previous simulation setup. We observe that as the minimum distance to the thresholds ($\min\{|x_n - l|, |u - x_n|\}$) increases, the constraint on the next sampling time that satisfies the missed detection probability requirement relaxes.

2.3.3.2 Extension to Non-Gaussian Innovations

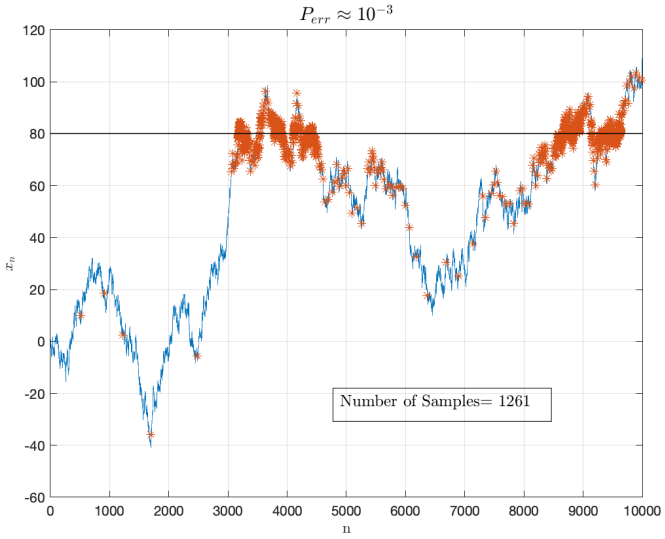
For a $p^{th}-$order process in (2.5), one needs to track densities at each time instant to calculate the probability of missing the threshold(s). However, by the Central Limit Theorem [13], $x[n]$ converges towards a standard Gaussian random variable in distribution as n approaches infinity. Hence, a practical approach to non-Gaussian innovations case could be tracking only a few densities, e.g., upto $n = 10$, and then assuming $x[n]$ follows a standard Gaussian random variable for $n > 10$.



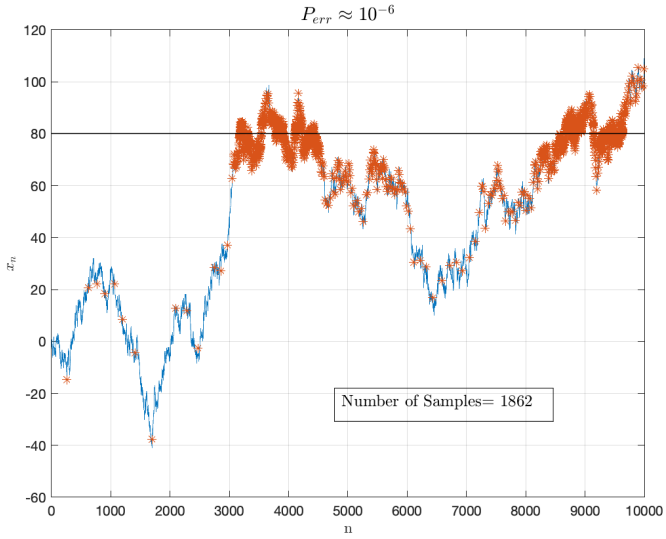
(a) Optimal sampling for at most 10^{-1} miss probability



(b) Optimal sampling for at most 10^{-2} miss probability

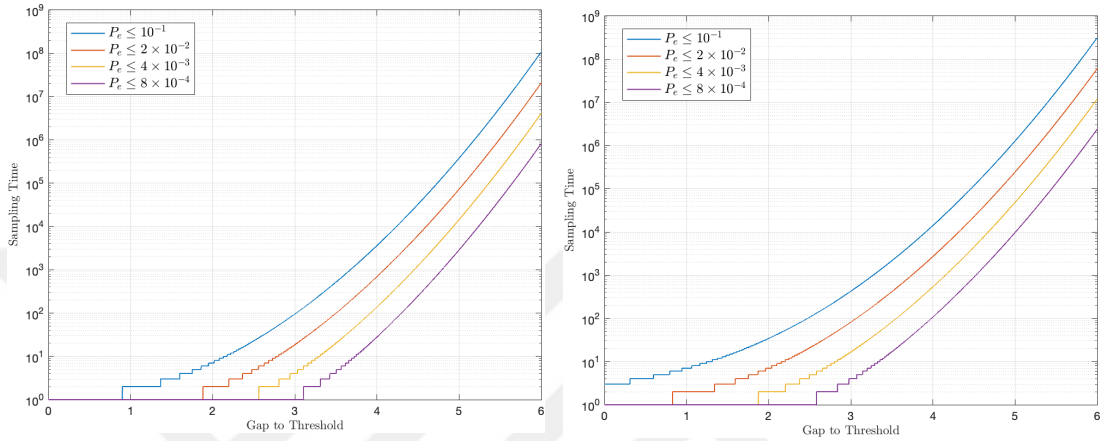


(c) Optimal sampling for at most 10^{-3} miss probability



(d) Optimal sampling for at most 10^{-6} miss probability

Figure 2.4: Optimal sampling strategy for various errors for Gaussian random walk



(a) $\alpha = \frac{1}{\sqrt{2}}$ (b) $\alpha = 0.99$

Figure 2.5: Optimal sampling time vs. gap to threshold for constant-variance AR(1) Gaussian process for various misdetection probabilities

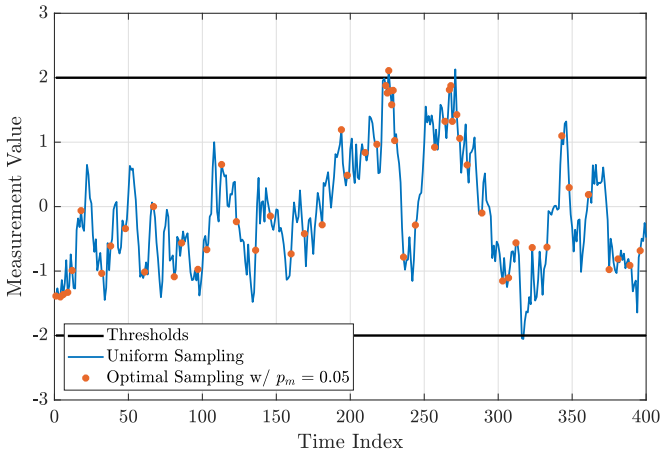
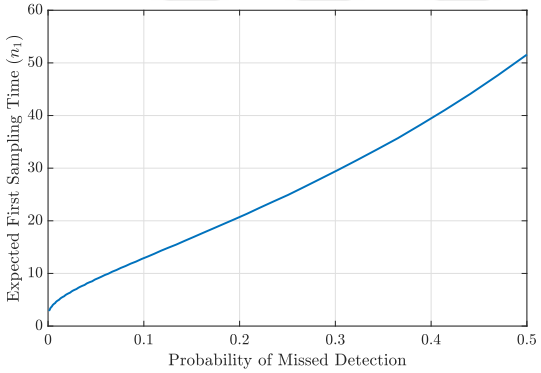
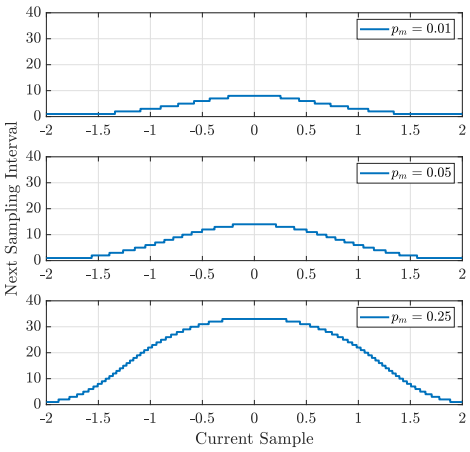


Figure 2.6: Optimal sampling strategy applied to an AR(1) process with double threshold.



(a) Expected first sampling time vs. probability of missed detection



(b) Next sampling time interval vs. current sample value for $p_m \in \{0.01, 0.05, 0.25\}$

Figure 2.7: Optimal sampling time vs. gap to threshold for constant-variance AR(1) Gaussian process for various misdetection probabilities

Chapter 3

Subspace Tracking in Semantic Signal Processing and Communication

3.1 Primer on Goal-Oriented Semantic Signal Processing and Communication

Recently a goal-oriented semantic signal processing framework has been proposed to extract and process semantic information generated at sensor nodes [3]. In this framework, the semantic information is organized using a flexible graph-based language in a structured and hierarchical way. The proposed structure is made up of bipartite graphs that include the identified signal *components*, and *predicates* that show the state or relationship of the detected components. Each node in the graphs may also include layers of numerical attributes with different levels of complexity. Although the aforementioned structure of the proposed language is fixed, the particular definitions of the components, predicates, and attributes are application-specific. Therefore, the structures are only as complex as the specific requirements of the implementation at the sensor/agent level and the specific

situation at the sensor site. In comparison to the NL-based semantic signal processing applications [14], the proposed structure for the semantic information offers a much more efficient and unambiguous representation while inherently providing privacy and secrecy through relatively abstract graph signals. Moreover, the additional computational cost of introducing graph signal processing algorithms is balanced by the inherent low-order graph signals generated by the proposed framework as they are generated within an application-specific and limited dictionary.

The proposed structure of the goal-oriented semantic signal processing framework at a sensor node is shown in Fig. 3.1. Briefly, a raw input signal generated by the sensor is preprocessed (e.g., through up/downsampling or Fourier transform) before *semantic extraction*, where the sensor output is mapped to the target semantic output. Semantic filtering and post-processing blocks are implemented in a goal-oriented manner to reduce the range of semantic outputs to the specific goals and perform additional processing (e.g., source coding), respectively. Finally, the resulting goal-filtered and compressed semantic output is either transmitted or stored. The most critical component in the entire diagram shown in Fig. 3.1 is the semantic extraction block, since accurate and reliable mapping of the raw signals into the semantic language affects the performance of the remaining blocks that follow in the semantic signal processing chain. Before we present the methods and algorithms to improve the accuracy and reliability of the semantic extractor, we first review the semantic language structure of [3].

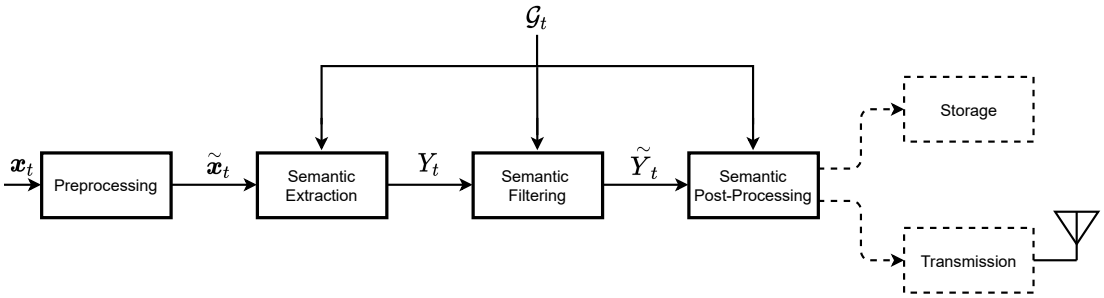


Figure 3.1: The proposed goal-oriented semantic signal processing framework [3]. Raw signal input is denoted as x_t, whereas the external goals are denoted with $\mathcal{G}_t$.

As detailed in [3], the semantic description of a signal starts with the definition

of component and predicate classes of interest as

$$C = \{c_1, c_2, \ldots, c_{N_c}\}, \tag{3.1}$$

$$P = \{p_0, p_1, p_2, \ldots, p_{N_p-1}\}, \tag{3.2}$$

where N_c and N_p are the numbers of component and predicate classes in the graph language, respectively. The semantic multi-graph description generated by the detected instances of C and P is defined as

$$\mathcal{D}_t = \{D_{t,1}, D_{t,2}, \ldots, D_{t,N}\}, \tag{3.3}$$

where each $D_{t,i}$ is an atomic bipartite graph that includes a connected set of detected component and predicate instances, as illustrated in Fig. 3.2. Note that the instance graph includes multiple instances of the same component or predicates; hence, each node in Fig. 3.2 is identified with a two-tuple, i.e., the component or predicate, and a unique ID number.

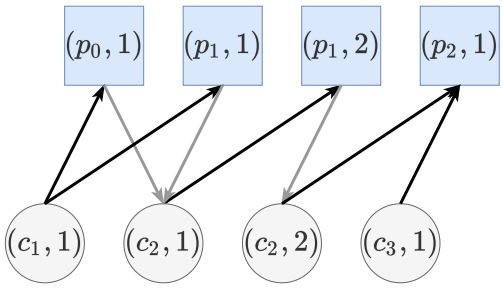


Figure 3.2: An example of an atomic bipartite graph structure ($D_{t,i}$) with signal components and predicates at the instance level. The two-tuple representation denotes the component or predicate class, and the unique instance identifier.

Each component and predicate in an atomic graph $D_{t,i}$ has a corresponding attribute set denoted by

$$\Theta_{t,i}(n_j, k) = \{\boldsymbol{\theta}_{t,i}^{(1)}(n_j, k), \boldsymbol{\theta}_{t,i}^{(2)}(n_j, k), \ldots, \boldsymbol{\theta}_{t,i}^{(L_{n_j})}(n_j, k)\}, \tag{3.4}$$

where (n_j, k) is the corresponding component or predicate node in the graph. Note that for each type of node n_j, we define L_{n_j}-levels of attributes. This enables an organized way of storing different attributes, preferably in increasing order of complexity for ease of goal-based filtering. We note that in the computer vision case study given in [3], there are three levels of attributes: the scalar position and

velocity in the lowest-level attribute set $\boldsymbol{\theta}_{t,i}^{(1)}$, subfeature vectors in $\boldsymbol{\theta}_{t,i}^{(2)}$, and the full bounding-box images of the detected components in $\boldsymbol{\theta}_{t,i}^{(3)}$. Nonetheless, the definition given in (3.4) is purposefully general, and can be modified depending on the specific application.

Recent advances in machine learning and artificial intelligence have enabled the extraction of semantic information in the proposed graph structure of (3.1)–(3.4). In practice, machine learning and artificial intelligence based semantic extractors do not have perfect accuracy or sensitivity; hence, the extracted semantic information does not have perfect correspondence with the objective reality. Moreover, the evolution of the generated semantic output must be tracked accurately to filter out any erroneous detection and identify the points of innovation so that the transmission (or storage) events can be scheduled efficiently.

An advantage of using learning-based algorithms is that statistical properties of the algorithms (e.g., confusion matrices) can be modeled through extensive training statistics, which can be used to improve the fidelity of the semantic output by exploiting the natural temporal continuity of the objects in the sensed area. Additionally, logical probabilities of the possible semantic output patterns can be exploited to improve reliability in a Bayesian sense, as well. For example, in a computer vision security application involving a car dealer showroom for a certain brand, identifying cars with other brands are much less likely logically, and thus, semantically. Therefore, the semantic extractor can take this into account during the temporal evolution of its semantic output to determine whether any new detections are the result of a true innovation or a sporadic erroneous event.

3.1.1 Semantic Modeling Example for One-Dimensional Signals

We can model the threshold detection problem studied in the previous subsection using the proposed semantic language introduced in Section 3.1. In a typical threshold detection problem, the semantic information extraction may include

the detection of the region in which the signal resides at a given moment, as well as the descriptions of threshold crossing events. Using the proposed formulation in Section 3.1, the components can be defined as the regions separated by the upper and lower thresholds that we denote as u and l.

$$c_1 = \{x \in \mathbb{R} \,|\, x > u\}, \tag{3.5}$$

$$c_2 = \{x \in \mathbb{R} \,|\, l \leq x \leq u\}, \tag{3.6}$$

$$c_3 = \{x \in \mathbb{R} \,|\, x < l\}. \tag{3.7}$$

We can easily extend the regions to a $k-$threshold model for $k > 2$ as well. Let $t_1 < t_2 < \ldots < t_k$ denote the thresholds in an ascending order. Then, the components can be defined as

$$c_1 = \{x \in \mathbb{R} \,|\, x > t_k\}, \tag{3.8}$$

$$c_i = \{x \in \mathbb{R} \,|\, t_{k-i+1} \leq x \leq t_{k-i+2}\}, \quad \forall i \in \{2, \ldots, k\}, \tag{3.9}$$

$$c_{k+1} = \{x \in \mathbb{R} \,|\, x < t_1\}. \tag{3.10}$$

The predicate set for this problem can be defined as descriptors of threshold crossing events as

$$P = (U, D, I), \tag{3.11}$$

where U, D, I denote *upward crossings*, *downward crossings*, and *idling* (staying in the same region), respectively. Note that for this particular case, the *class multi-graph description* $\mathcal{S}_t$ and the *instance multi-graph description* $\mathcal{D}_t$ are defined as equivalent, since a higher level abstraction of a scalar signal is not required.

For the multi-level attribute set, a suitable choice for the first level is the current time index and amplitude of the process, and for each level l we can store the previous $(l-1)$-th time and amplitude information as

$$\mathcal{A}_t = \{\Theta_{t,i}\} \text{ where } \Theta_{t,i} = \{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(r)}\}, \tag{3.12}$$

$$\theta^l = (n - l + 1,\, x_{n-l+1}), \tag{3.13}$$

where r is the total number of levels in the attribute sets. Depending on the specific application, alternative attribute levels including desired transform domain representations of the signal can be defined.

For a two-threshold case, and using the same Gaussian AR(1) process input signal in (2.25) with $l = -2$ and $u = 2$, we illustrate a region of interest around sampling indices $\tilde{n} \in [38, 45]$ in Fig. 3.3. Note that sampling points $\tilde{n}$ are not the same as the time index n, as we employ the optimal sampling strategy described in Section 2.3 to improve the energy and processing efficiency of the sensor. In Fig. 3.3, the samples marked with circles, upward-pointing triangles, and downward-pointing triangles represent *idling* (I), *upward crossing* (U), and *downward crossing* (D) events, respectively. In Fig. 3.4, class-level graph representations of the samples in Fig. 3.3, at $\tilde{n} = 41$ and $\tilde{n} = 42$, corresponding to $n = 225$ and $n = 226$ in Fig. 2.6, are illustrated with the previously defined semantic graph representation in (3.5)–(3.11). Additionally, for the semantic graphs shown in Fig. 3.4, the attribute sets as defined in (3.12), and (3.13) for $r = 2$ are listed in Table 3.1.

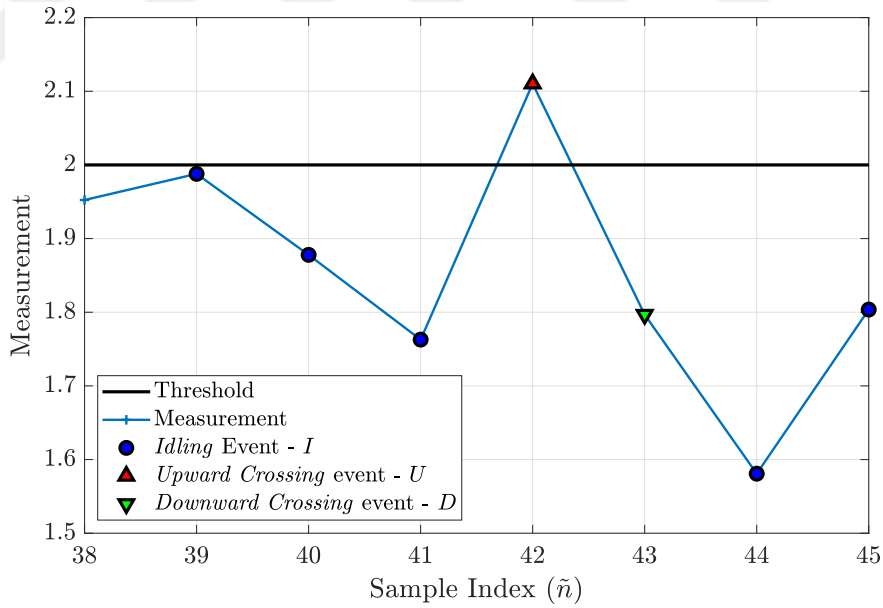
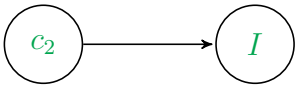
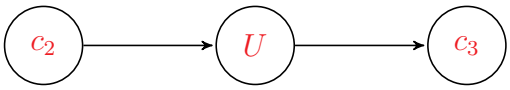


Figure 3.3: Illustration of *idling*, *upward crossing*, and *downward crossing* events for the AR(1) signal given in Fig. 2.6.

As seen in Fig. 3.3, the signal is *idling* in Region-2 (between thresholds) at $\tilde{n} = 41$; hence, the corresponding component is c_2 and its predicate is I. At $\tilde{n} = 42$, the signal exceeds the upper threshold; therefore, the semantic description includes a flow from c_2 to c_3 with the predicate U (*upward crossing*).



(a) Semantic graph for $\tilde{n} = 41\,(n = 225)$.



(b) Semantic graph for $\tilde{n} = 42\,(n = 226)$.

Figure 3.4: Semantic graph descriptions for the events depicted in Fig. 3.3, illustrating the *idling* and *upward crossing* predicates.

	$\tilde{n} = 41\,(n = 225)$	$\tilde{n} = 42\,(n = 226)$
θ^1	(225, 1.76)	(226, 2.11)
θ^2	(224, 1.88)	(225, 1.76)

Table 3.1: The attribute sets for samples taken for time instants 225 and 226, generated as companions to graphs given in Fig. 3.4.

With the adoption of the semantic language in Section 3.1, we can easily define specific goals to reduce the storage or transmission rates for this particular application. In a $k-$threshold crossing detection problem, a typical goal might be the detection of exceeding the largest threshold, i.e., $x > t_k$. This means that we are interested in a *upward crossing* predicate connected to c_k, hence the goal pattern can be defined as shown in Fig. 3.5.

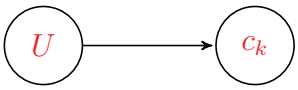


Figure 3.5: Goal pattern defined for an interest in detecting largest threshold crossings.

With the goals as defined in Fig. 3.5, the transmission events (or, storage events for offline applications) can be reduced considerably. Coupled with the introduction of optimal sampling strategies in Section 2.3, the overall energy and processing efficiency of scalar sensors can be improved dramatically.

3.2 Subspace Tracking of Vector Attributes

In the semantic signal processing framework of [3] summarized in Section 3.1 and illustrated in Fig. 3.1, the semantic extractor is assumed to be generating reliable and smooth semantic output signals. On the other hand, the extracted semantic information includes semantic noise that is introduced by input noise, algorithm limitations, etc. In this section, we propose methods that can be used to exploit prior information about the semantic extractor and environmental characteristics to improve the fidelity of the extracted semantic information.

The conceptual block diagram for a semantic extractor is given in Fig. 3.6. Note that the *Raw Semantic Extraction* block typically involves a learning-based algorithm such as a scene-graph generator for image/video signals [15], segmented and classified objects in LIDAR images [16], sound event detection (SED) for acoustic signals [17], etc. The following blocks in Fig. 3.6 exploit the known characteristics of the raw extractor, such as its confusion matrix; and the statistics of the environment, such as the occurrence probabilities of the possible output signals. Throughout this section, we focus on the *Attribute tracking* module.

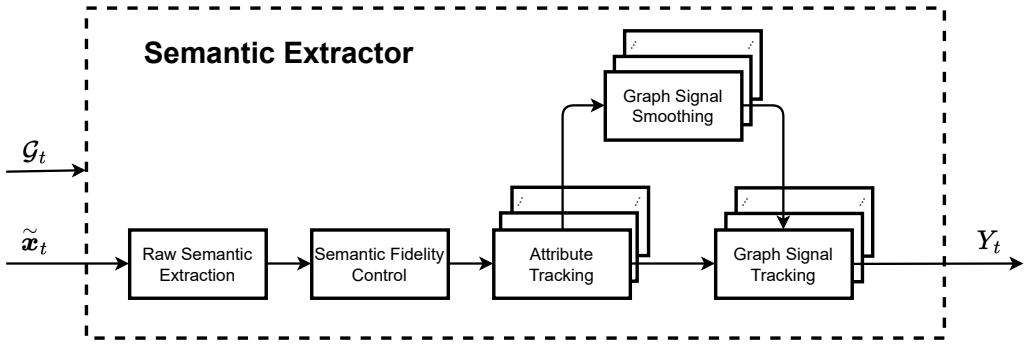


Figure 3.6: Proposed block diagram for a typical graph-based semantic extractor.

The *Semantic Fidelity Control* block takes the raw output of the initial extraction and uses time integration techniques to improve the detection characteristics according to internally set or externally requested reliability parameters.

Note that after this point in the semantic extraction framework shown in Fig. 3.6, tracking and smoothing blocks work in parallel processing banks, with

each instance working on an atomic graph as defined in (3.3). This parallel approach is the result of intentional separation of the inter-connected atomic graphs, and reduces the computational complexity of the following blocks.

In a typical application of semantic signal processing or communications, the innovation in the semantic signals will not only come from the graph structure, but also from the low-level numerical attributes embedded in the graphs (see (3.4) in Section 3.1). The *Attribute Tracking* is an application-specific block, where these low-level numerical attributes can be tracked across time to detect innovations in the semantic signals of interest. For vector attributes (e.g., subfeature vectors or state vectors), we propose and demonstrate a subspace tracking algorithm in Section 3.2.1 and 3.2.2.

The *Graph Signal Smoothing* block is proposed as an optional pre-filtering step to smooth out sporadic erroneous detections and reduce the throughput for the following blocks. The final block in Fig. 3.6, the *Graph Signal Tracking*, uses the pre-filtered graph signal and attribute updates in conjunction with the estimated environmental probabilities to produce a reliable, accurate semantic description of the input signal.

3.2.1 Subspace Tracking and Robust PCA Algorithms

In the raw semantic extraction block shown in Fig. 3.6, the semantic data are organized to include goal-filtered components and their corresponding attributes. In a typical computer vision problem, these attributes can be position, speed for scalar attributes, sub-feature vectors for vector attributes, or even the encoded video signal itself. Tracking these attributes enables identification of strong changes at the attribute level, which in turn can be used to update the transmitted or stored attributes, even when the overarching semantic graph structure stays the same. On the other hand, if the semantic graph signal is erroneously updated (or kept the same), attribute tracking may be used to reconcile consecutive attributes to detect and correct these errors.

We now present a subspace tracking algorithm that can be used for vector attributes. To illustrate the algorithm more clearly, we use the real-world semantic extraction of video-streams presented in [3] where we define signal components as detected objects in the stream.

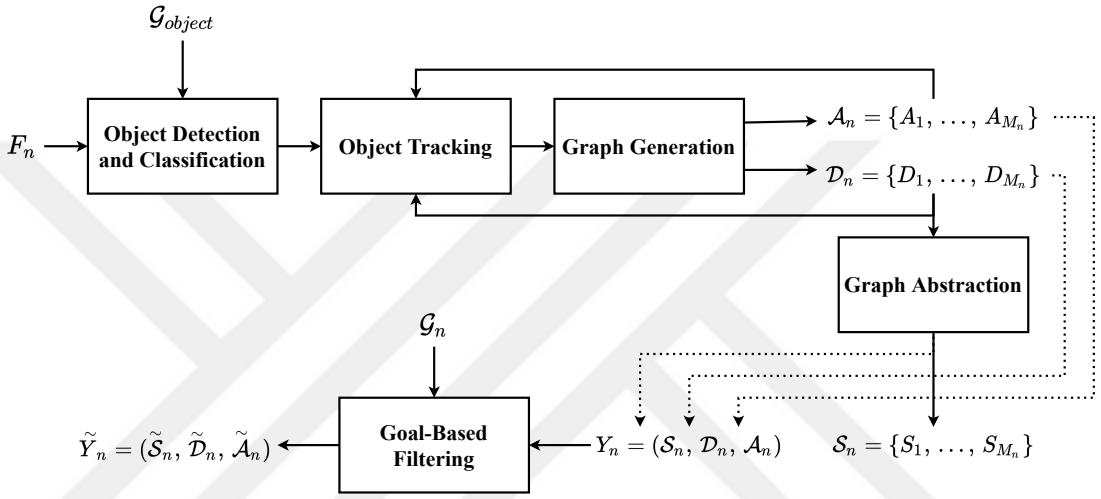


Figure 3.7: Semantic extraction for real-time video signals as presented in [3].

As illustrated in Fig. 3.7, the proposed semantic extractor in [3] utilizes YOLOv4-CSP [18] model to detect occurring objects in each input frame F_n. For temporal extension, we adopt *tracking by detection* paradigm and utilize DeepSORT [19] to track individual objects in the video stream which utilizes Kalman filter to recursively predict future positions of detected objects. Furthermore, DeepSORT inherits another small convolutional neural network model trained on re-identification task to extract visual feature vectors of each detected object to be used for association of existing tracks and recent detections.

We refer readers to [19, 20, 21, 22] for details about tracking by detection paradigm. We denote these feature vectors with $\boldsymbol{r}_i \in \mathbb{R}^{128}$, where $||\boldsymbol{r}_i||_2 = 1$, and utilize them as second-level vector attributes of component nodes in our semantic graph output (see [3] for details). We would like to point out that the feature vector $\boldsymbol{r}_i$ for a particular detected component is a 128-dimensional vector and carries most of the attribute-level *innovation* in the semantic signal processing framework.

In Fig. 3.7, the object-level goal $\mathcal{G}_{object}$ restricts the components and the predicates that must be detected. The corresponding *multi-graph instance representation* $\mathcal{D}_n = \{D_1, \ldots, D_{M_n}\}$ composed of M_n disconnected subgraphs and a corresponding attribute set $\mathcal{A}_n = \{A_1, \ldots, A_{M_n}\}$ extracted from the video signal. The *multi-graph class representation* $\mathcal{S}_n$ is constructed in the Graph Abstraction block, and the complete semantic description $Y_n = (\mathcal{S}_n, \mathcal{D}_n, \mathcal{A}_n)$ is generated. Finally, if there are external goals $\mathcal{G}_n$, the semantic description is filtered to obtain the goal-filtered semantic description.

To process and track attribute vectors efficiently, we use dimensionality reduction techniques such as Robust PCA and subspace tracking [23]. To track the evolution of the subspace of feature vectors with respect to time, we use Fast Principal Component Pursuit (PCP) via alternating minimization [24].

Defining a feature matrix D, with its rows being the subfeature vectors across time, and decomposing the matrix into a compressed basis matrix L and residual matrix S, the PCP problem is formulated as

$$\arg\min_{L,S} \|L\|_* + \lambda\|S\|_1 \quad \text{subject to} \quad D = L + S, \tag{3.14}$$

where $\|L\|_*$ is the nuclear norm of matrix L and $\|S\|_1$ is the l_1-norm of matrix S. A variant of (3.14) can be constructed as

$$\arg\min_{L,S} \frac{1}{2}\|L + S - D\|_F + \lambda\|S\|_1 \quad \text{subject to} \quad \text{rank}(L) = t, \tag{3.15}$$

where $\|L + S - D\|$ denotes the Frobenius norm of $L + S - D$. Finally, we can solve the problem via alternating minimization, namely,

$$L_{k+1} = \arg\min_{L} \|L + S_k - D\|_F \text{ subject to } \text{rank}(L) = t \tag{3.16}$$

$$S_{k+1} = \arg\min_{S} \|L_{k+1} + S - D\|_F + \lambda\|S\|_1. \tag{3.17}$$

The only computationally demanding step is (3.16), and it can be solved by computing a partial SVD of $D - S_k$ with t components. The solution to (3.17) is *element-wise shrinkage*, i.e., $\text{sign}(D - L_{k+1})\left(|D - L_{k+1}| - \lambda\right)^+$, where $(x)^+ = \max\{0, x\}$. Note that t in (3.16) can be chosen by a heuristic procedure.

We increment t and estimate the contribution of the new singular vector. Comparing the contribution with a threshold, the algorithm stops at a particular t. In numerical simulations, we observed that t is usually small (does not exceed 3 in our setting).

3.2.2 Use of Subspace Tracking in Semantic Post-Processing

To utilize PCP to track the evolution of the feature vectors in a practical application, one can solve the alternating minimization problem given in (3.16) and (3.17) periodically on a moving window. The size of this window (buffer) depends on the application and the rate of change in the semantic content. Then, a simple heuristic approach can be taken to detect innovations by checking the l_1-norm of the sparse component S after convergence and compare it with a predefined threshold.

3.2.3 Real Time Video Processing Case Studies

To illustrate the inherent redundancy for consecutive frames, we show the still images from a 10s video that includes a car passing from a shaded region to a sunny region in Fig. 3.8. The correlations of consecutive attribute vectors are given in Fig. 3.12(a). Note that in the scenario given in Fig. 3.8, the semantic components and predicates stay the same (i.e., the same objects and relationships are present throughout the clip), while the attributes of the car component (car-3) change due to the image brightness, contrast, etc.

3.2.3.1 Case Study I: Car in a Parking Lot

To illustrate the vector subspace tracking method developed in Section 3.2.1 and 3.2.2, we use the same recorded video example shown in Fig. 3.8. Running the

Figure 3.8: A car is passing from a shaded region to a sunny region. The middle and right sub-figures demonstrate the *innovation* event at the attribute level, while the semantic components stay the same.

PCP algorithm given in (3.14)–(3.17) on the video clip, we obtain the subspace breakdown of vector attributes shown in Fig. 3.9.

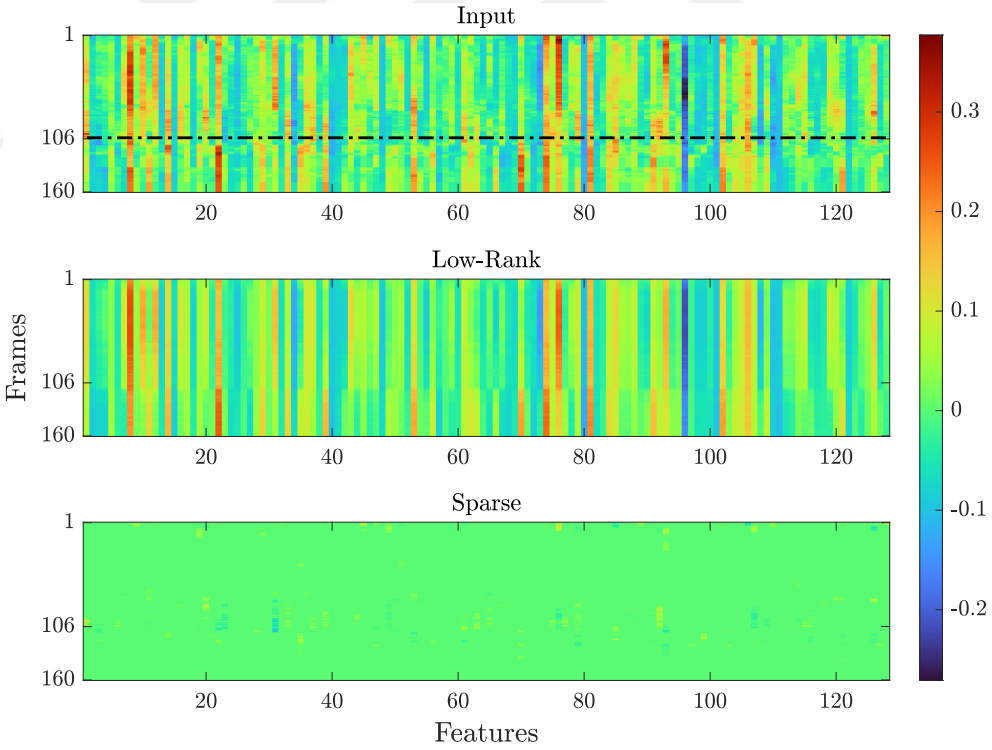


Figure 3.9: PCP output of the car video

As it is seen in the sparse component plot, we have two innovations, i.e., when the car first appears in the scene and when it passes to the shaded region (around frame 106). By plotting the l_1-norm of the sparse component, we can track the innovation versus time.

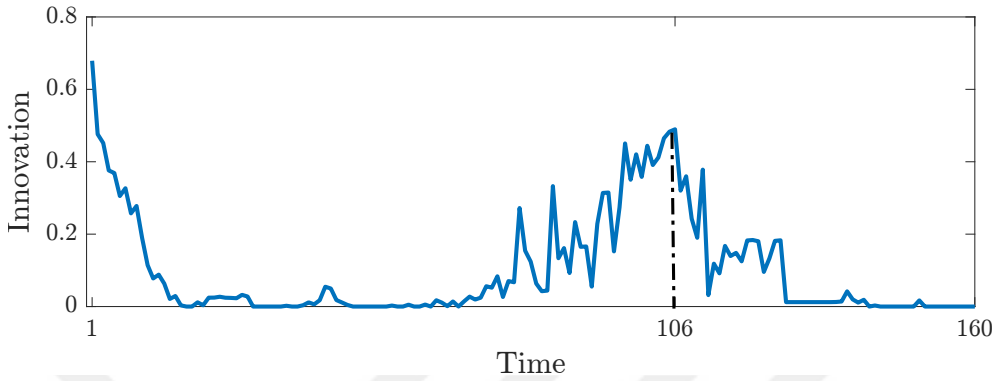


Figure 3.10: Innovation vs. time for the car video

To design an attribute tracking algorithm that can work in real-time, one can use fixed-sized buffers and run the algorithm continuously. Again, we can track the l_1-norm of the sparse components to detect an attribute-level innovation in the scene, as shown in Fig. 3.10. Moreover, Fig. 3.11 illustrates the effect of buffer size on the innovation detection. As expected, smaller buffer sizes result in highly concentrated, spike-shaped innovations. Furthermore, the innovation at the beginning diminishes as the buffer size gets smaller.

As shown in Fig. 3.12(a), an innovation at the attribute level can be identified around frame-106, when the lighting changes for the car-3 component in the overall semantic graph signal. In Fig. 3.12(b), the attribute level innovation has been illustrated by analyzing the l_1-norm of the sparse component of the feature matrix D. As the norm is maximum at frame 106, we identify the innovation and locate the precise time instance quantitatively.

To illustrate the inherent redundancy for consecutive frames, we show the still images from a video that includes a pedestrian passing from a shaded region to a sunny region in Fig. 3.13. The correlations of consecutive attribute vectors are given in Fig. 3.16(a). Note that in the scenario given in Fig. 3.13, the semantic components and predicates stay the same (i.e., the same objects and relationships are present throughout the clip), while the attributes of the pedestrian component change due to the image brightness, contrast, etc.

As shown in Fig. 3.16(a), an innovation at the attribute level can be identified

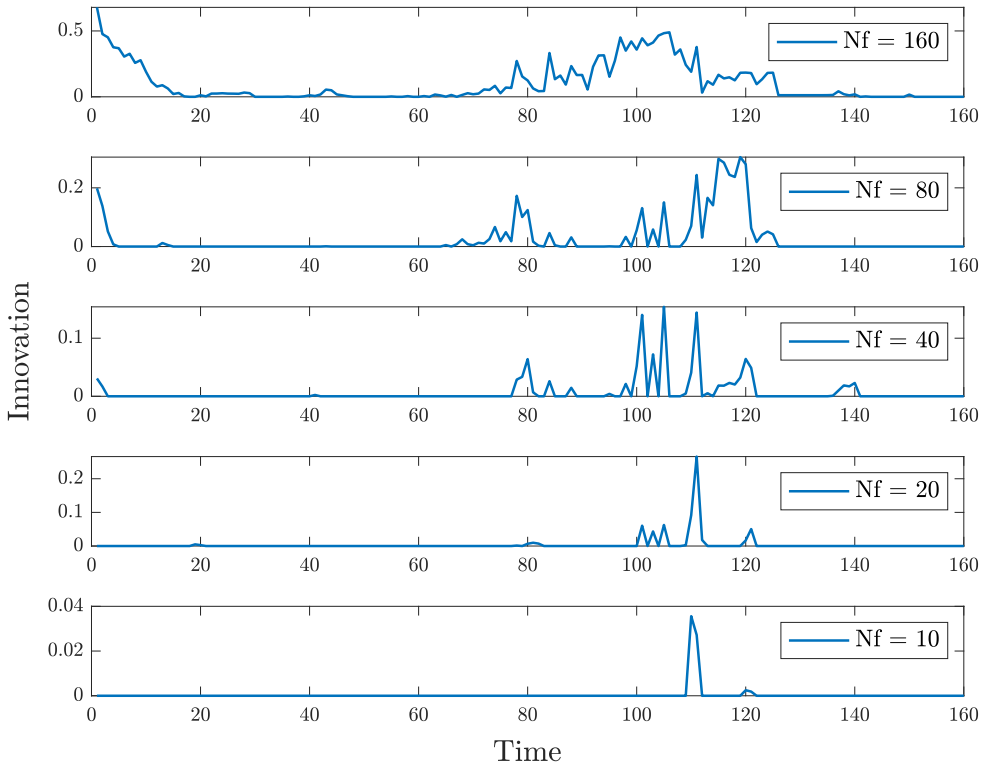


Figure 3.11: Innovation vs. time for various buffer sizes for the car video

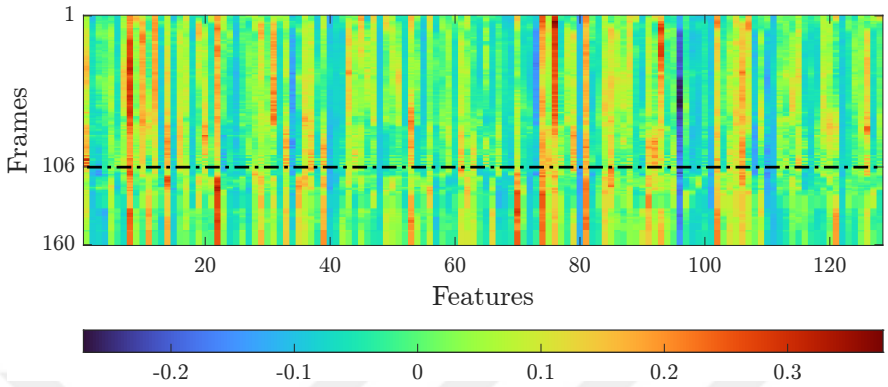
around frame-106, when the lighting changes for the car-3 component in the overall semantic graph signal.

3.2.3.2 Case Study II: Pedestrian in a Parking Lot

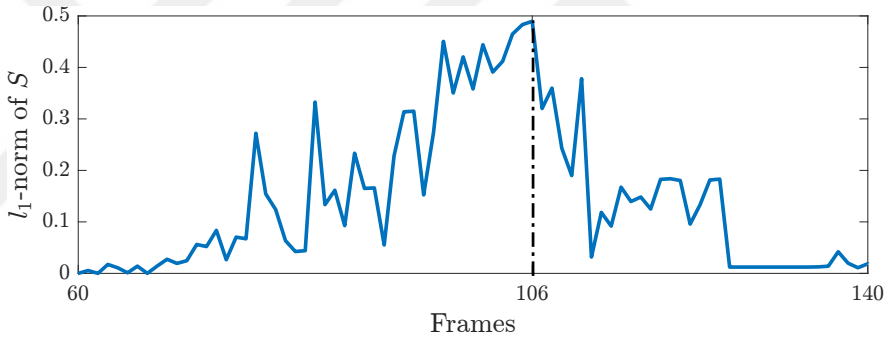
Next, we repeat the analysis for a pedestrian walking straight from a shaded region to light region as shown in still images from a video in Fig. 3.13. Running the PCP algorithm given in (3.14)–(3.17) on the video clip, we obtain the subspace breakdown of vector attributes shown in Fig. 3.14.

As it is seen in the sparse component plot, we have an innovation, i.e., when the pedestrian passes to the shaded region (around frame 315). By plotting the l_1-norm of the sparse component, we can track the innovation versus time.

To design an attribute tracking algorithm that can work in real-time, one can



(a) Correlation of subfeature vectors across consecutive frames. The innovation at the attribute level is illustrated by a black dashdotted line.



(b) Frames vs. l_1-norm of the sparse component S. Note that the attribute level innovation is visible around the frame-106 since the l_1-norm of the sparse component achieves its maximum.

Figure 3.12: Attribute level innovation analysis for the car example given in Fig. 3.8

use fixed-sized buffers and run the algorithm continuously. Again, we can track the l_1-norm of the sparse components to detect an attribute-level innovation in the scene, as shown in Fig. 3.15. In Fig. 3.16(b), the attribute level innovation has been illustrated by analyzing the l_1-norm of the sparse component of the feature matrix D. As the norm is maximum at frame 315, we identify the innovation and locate the precise time instance quantitatively.

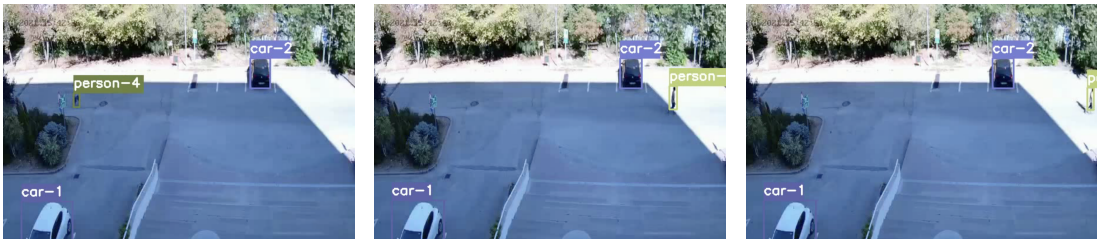


Figure 3.13: A pedestrian is passing from a shaded region to a sunny region. The middle and right sub-figures demonstrate the *innovation* event at the attribute level, while the semantic components stay the same.

3.2.3.3 Case Study III: Object Reconciliation

In certain scenarios where multiple instances of an object are classified with unique identifiers (ID), the same instance of the class can be detected; however, ID numbers can change due to occlusions, buffer sizes, or missed detections. We demonstrate that even in these cases, the proposed method is able to reconcile the different identifications.

In Fig 3.17, an object detected with multiple IDs is shown throughout the video. Using the PCP algorithm, we analyze the low rank components of these seemingly separate IDs in Fig 3.18. To reconcile these objects, we calculate the mean of low-rank components over time and compare the Manhattan distance between them, e.g., $\|L_7 - L_1\|_1$ vs. $\|L_7 - L_2\|_1$ to reconcile the instance with ID 7. The resulting distance comparison is given in Fig 3.19. As illustrated, IDs 7, 16, and 17 are correctly reconciled with the instance ID 2.

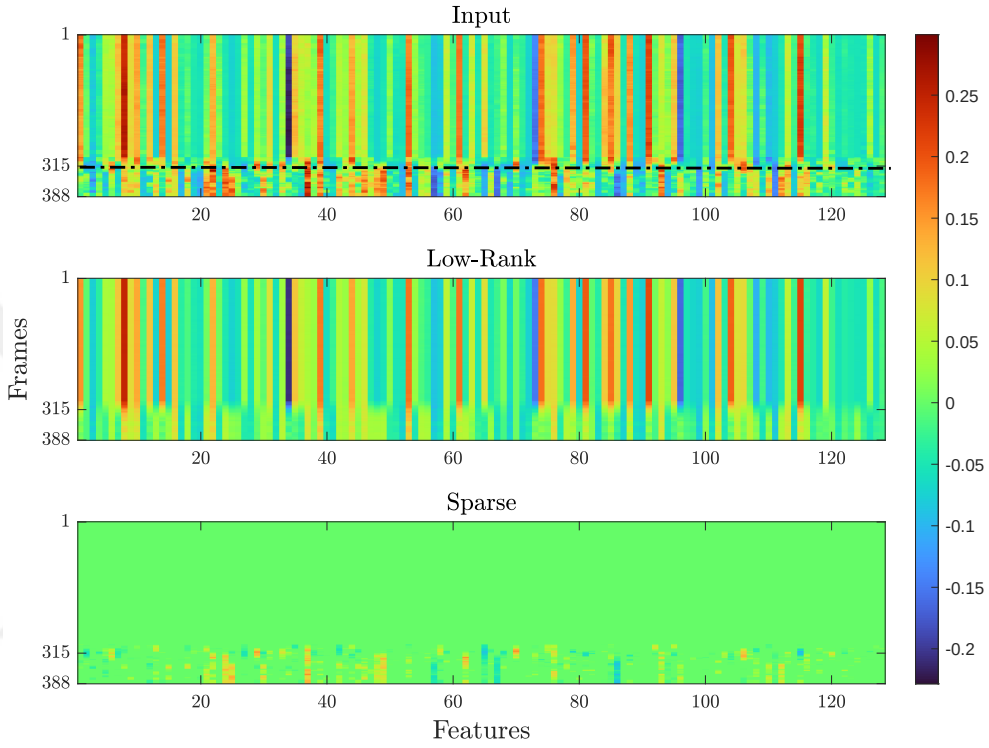


Figure 3.14: PCP output of the pedestrian video

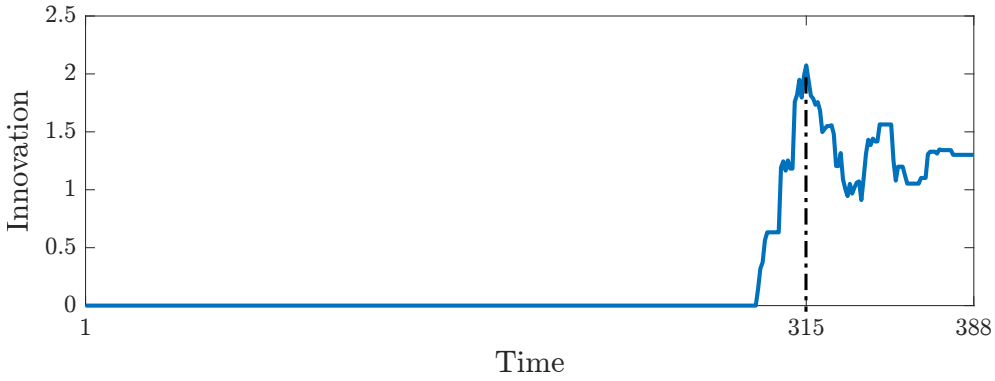
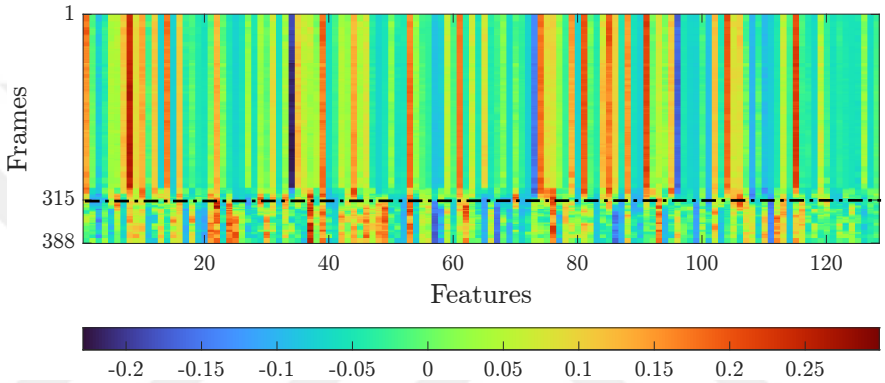
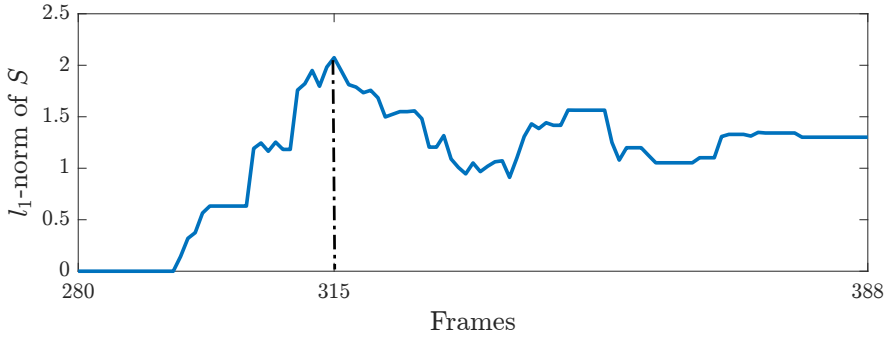


Figure 3.15: Innovation vs. time for the pedestrian video



(a) Correlation of subfeature vectors across consecutive frames. The innovation at the attribute level is illustrated by a black dashdotted line.



(b) Frames vs. l_1-norm of the sparse component S. Note that the attribute level innovation is visible around the frame-315 since the l_1-norm of the sparse component achieves its maximum.

Figure 3.16: Attribute level innovation analysis for the pedestrian example given in Fig. 3.13

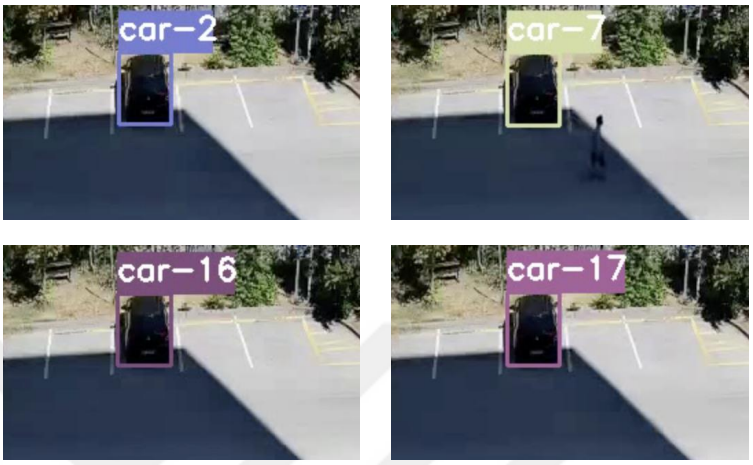


Figure 3.17: An object is assigned different IDs during the video

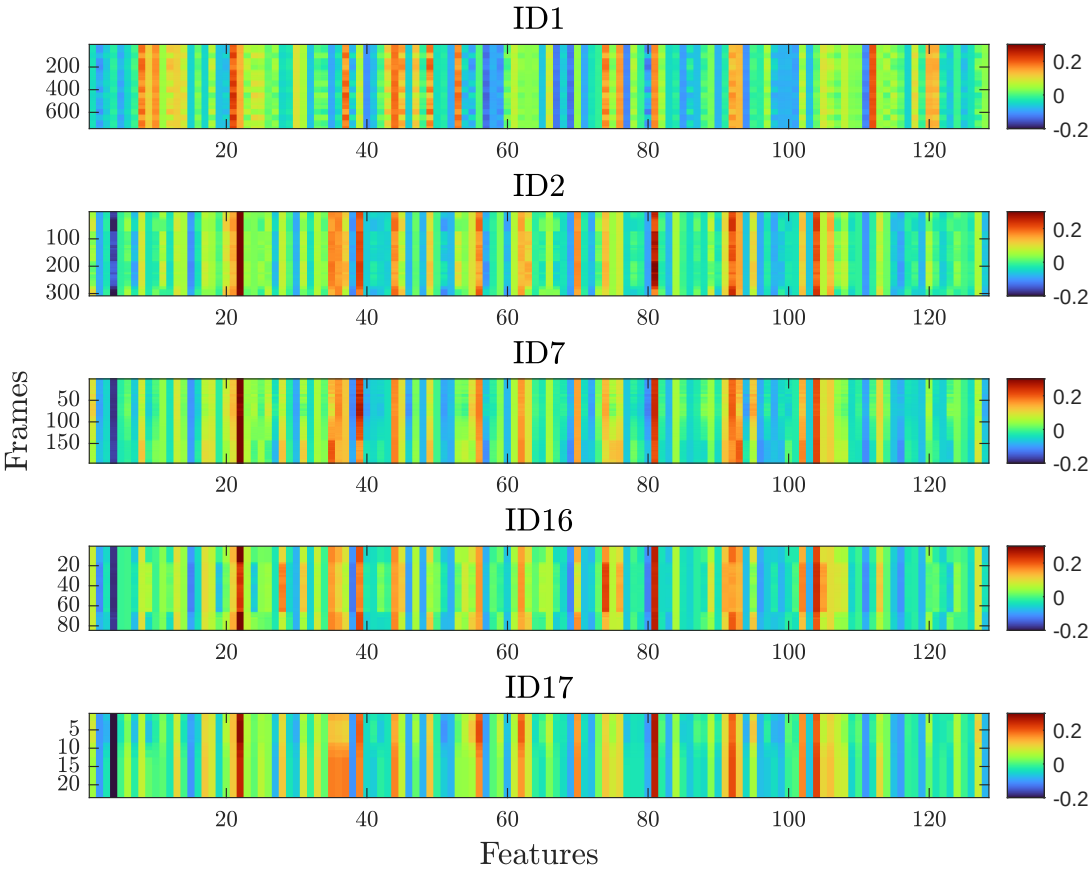


Figure 3.18: Low-rank features vs. time different IDs

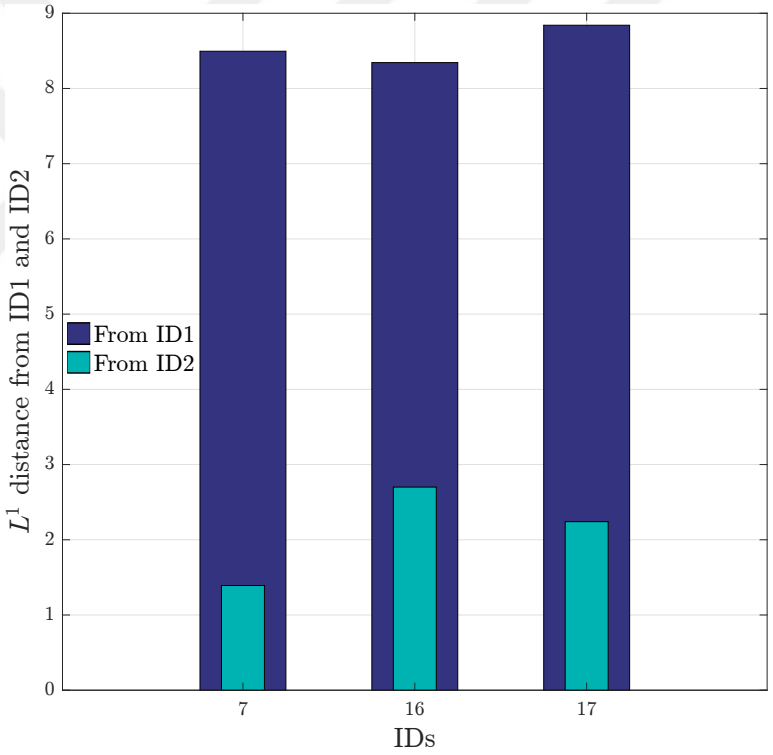


Figure 3.19: Innovation vs. time for various buffer sizes for the car video

Chapter 4

Conclusions and Future Work

Since the number of connected IoT devices increases massively, efficiency plays a vital role in the next generation systems. In this thesis, the energy efficiency is investigated for goal-oriented IoT sensor networks and semantic signal processing and communication applications.

The analyses carried on show that certain tasks can be implemented in a goal-oriented setting more efficiently than using off-the-shelf methods, and semantic processing can benefit significantly from the existing dimensionality reduction schemes which allow identification of significant innovations in the semantic signal by exploiting known semantic characteristics of the environment.

The proposed methods provide reliable semantic outputs and enable efficient ways of identifying semantic innovation, while filtering out unwanted semantic noise. Note that the proposed metrics and methods can also be used to schedule transmission and storage events in semantics-enabled sensor devices.

As the innovation detection proposed in this work and in the literature move closer to massive deployment in the next generation of sensor networks, continued research on the practical implications of the proposed methods is required. Specifically, the proposed method has to be adapted to other sensor modalities and multi-sensor fusion for detection of innovation should be addressed.

We finally note that, depending on the type of sensor/device and its computational capabilities, the proposed method can be adapted. As the signal processing and communications paradigms move towards semantic signal processing and transmission, we believe the proposed semantic innovation detection framework will be an essential building block in developing the next generation of sensor devices and networks.

Bibliography

[1] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information systems frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[2] M. Hasan, "State of IoT 2022: Number of connected IoT devices growing 18

[3] M. Kalfa, M. Gok, A. Atalik, B. Tegin, T. M. Duman, and O. Arikan, "Towards goal-oriented semantic signal processing: Applications and future challenges," *Digital Signal Processing*, vol. 119, p. 103134, 2021.

[4] D. Giusto, A. Iera, G. Morabito, and L. Atzori, *The internet of things: 20th Tyrrhenian workshop on digital communications*. Springer Science & Business Media, 2010.

[5] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[6] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Transactions on industrial informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[7] S. Li, L. Da Xu, and S. Zhao, "5G Internet of Things: A survey," *Journal of Industrial Information Integration*, vol. 10, pp. 1–9, 2018.

[8] "5G and IoT: The mobile broadband future of IoT," Apr 2022.

[9] L. Zhang, Y.-C. Liang, and D. Niyato, "6G visions: Mobile ultra-broadband, super internet-of-things, and artificial intelligence," *China Communications*, vol. 16, no. 8, pp. 1–14, 2019.

[10] R. G. Gallager, *Stochastic Processes: Theory for Applications*. Cambridge University Press, 2017.

[11] K. B. Erickson, "Strong renewal theorems with infinite mean," *Transactions of the American Mathematical Society*, vol. 151, no. 1, p. 263–263, 1970.

[12] A. A. Markov, *Theory of algorithms*. Academy of Sciences of the USSR, 1954.

[13] R. Durrett, *Probability: Theory and examples*. Cambridge University Press, 2020.

[14] J. Bao, P. Basu, M. Dean, C. Partridge, A. Swami, W. Leland, and J. A. Hendler, "Towards a theory of semantic communication," in *Proceedings of the 2011 IEEE 1st International Network Science Workshop*, pp. 110–117, 2011.

[15] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Graph R-CNN for scene graph generation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 670–685, 2018.

[16] E. Che, J. Jung, and M. J. Olsen, "Object recognition, segmentation, and classification of mobile laser scanning point clouds: A state of the art review," *Sensors*, vol. 19, no. 4, 2019.

[17] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021.

[18] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: Scaling cross stage partial network," in *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pp. 13029–13038, 2021.

[19] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3645–3649, IEEE, 2017.

[20] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, "BoT-SORT: Robust associations multi-pedestrian tracking," *arXiv preprint arXiv:2206.14651*, 2022.

[21] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang, "ByteTrack: Multi-object tracking by associating every detection box," *arXiv preprint arXiv:2110.06864*, 2021.

[22] Y. Du, Y. Song, B. Yang, and Y. Zhao, "StrongSORT: Make DeepSORT great again," *arXiv preprint arXiv:2202.13514*, 2022.

[23] N. Vaswani, T. Bouwmans, S. Javed, and P. Narayanamurthy, "Robust subspace learning: Robust PCA, robust subspace tracking, and robust subspace recovery," *IEEE Signal Processing Magazine*, vol. 35, no. 4, pp. 32–55, 2018.

[24] P. Rodríguez and B. Wohlberg, "Fast principal component pursuit via alternating minimization," in *2013 IEEE International Conference on Image Processing*, pp. 69–73, 2013.