Journal of Real-Time Image Processing (2022) 19:445–461
https://doi.org/10.1007/s11554-022-01198-z

ORIGINAL RESEARCH PAPER

Temporal synchronization framework of machine-vision cameras for high-speed steel surface inspection systems

Vasanth Subramanyam[1,2] · Jayendra Kumar[2] · Shiva Nand Singh[2]

Received: 10 June 2021 / Accepted: 31 December 2021 / Published online: 27 January 2022

Abstract
High-speed industrial machine-vision (MV) applications such as surface inspection of steel sheets necessitate synchronous operation of multiple high-resolution cameras. Synchronization of cameras in the microsecond band is necessary to ensure accurate frame matching while melding images together. Existing approaches for synchronization employ dedicated electronic circuits or network-time-protocol (NTP) whose accuracies are in the millisecond band. Conversely, IEEE-1508 precision-time-protocol (PTP) synchronizes computers in highly accurate industrial measurement and control networks. Synchronization algorithms using PTP involve synchronizing computers connected to cameras. Although the computers synchronize in the microsecond band, the cameras synchronize in the millisecond band. Moreover, PTP is practically not used for synchronizing multiple devices due to the high bandwidth utilization of the network. This paper proposes a temporal synchronization algorithm and framework with two-way communication with timestamps and estimates mean path delays. Unicast transmission forms the basis of the synchronization framework, so that the network utilization is minimal, thereby ensuring the necessary bandwidth is available for image transmission. Experimental results show that the proposed approach outperforms the existing methodologies with synchronization accuracies in the microsecond band.

Keywords Frame synchronization · Machine vision · Manufacturing automation

1 Introduction

Important domains in the manufacturing sector, such as measurement, instrumentation, quality control, material tracking, defect detection, defect classification, etc., employ state-of-the-art technologies such as machine-vision [1] owing to their high accuracy and precision. Such industrial applications require high-resolution cameras with lenses that are selected based on the application [2].

The major limitations of such systems are the limited field-of-view (FOV) of cameras that are solved using two methods. The first method involves using omni-directional cameras [3] that employ reflective mirrors arranged specifically to enhance the FOV. The disadvantage of this method is that the resolution of the resultant image is not uniform. The second method involves using multiple cameras arranged in a predefined scheme known as distributed aperture systems (DAS) [4]. These cameras provide a dynamic view of their immediate surroundings, thereby producing broader FOV images.

Many versions of DAS systems [5, 6] are reported in the literature that employs high-resolution IR sensors to provide complete 360° views. Apart from these inventions, there have been other applications involving robotic vehicles [7] comprising multiple visible and IR cameras to provide simultaneous views.

The motivation for this paper arises from scenarios of continuous web processes such as steel rolling mills [8, 9], fabric mills where the objective of the DAS is capturing surface defects, as well as the location of defects in real time [10, 11]. Although there are many publications regarding the synchronization of multiple devices, this paper focuses on machine-vision cameras specifically for the following reasons:

- The tolerance in synchronization accuracies of machine-vision cameras is least, since the viewpoint captured by

✉ Vasanth Subramanyam
v.subramanyam@tatasteel.com

[1] Tata Steel Ltd, Jamshedpur 831001, India

[2] Department of ECE, National Institute of Technology, Jamshedpur 831014, India

the multiple devices should be same in fast changing industrial environments.
- The bandwidth requirement for transmission of images is very high; hence, the bandwidth consumed by the synchronization methodology should be minimal.

The main contributions of this paper are as follows:

- This paper presents a novel synchronization algorithm for two sensors. In this regard, the algorithm estimates the relative clock offset and skew and also provides a methodology for two-way communication of synchronization signals (inspired by PTP).
- This paper extends the synchronization algorithm to propose a novel synchronization framework for a network of sensors. The framework aids in the addition and removal of new sensors in the network without additional configuration.
- The proposed synchronization algorithm is compared with existing approaches in the literature that use NTP and PTP. The experimental results are presented and discussed.
- The validation of the proposed algorithm and framework is presented and the results are discussed.

Although the synchronization algorithm and framework proposed in this paper can be applied to any sensors requiring synchronization, this paper utilizes machine-vision cameras as sensors for all practical purposes, since the paper is motivated from surface inspection systems in the steel industry that employ machine-vision cameras.

The organization of the rest of the paper is as follows: Sect. 2 discusses the motivation for proposing a synchronization mechanism and usage of machine-vision cameras as the sensors needing synchronization; Sect. 3 surveys the related work and introduces the research gap; Sect. 4 states the problem mathematically; Sect. 5 proposes a synchronization algorithm for two sensors; Sect. 6 proposes a framework to synchronize a network of multiple sensors; Sect. 7 provides the hardware setup utilizing machine-vision cameras as sensors; Sect. 8 describes experimental results; Finally, Sect. 9 concludes the paper and provides probable direction for future intended research.

2 Motivation

The motivation for this paper can be understood through the industrial use-case scenario presented, and machine-vision cameras inspect the surface of steel sheets for identification and qualification of defects, as shown in Fig. 1.

Steel rolling processes are classified into hot and cold rolling. Hot rolling involves heating a steel slab and then flattening it into thinner sections of strips. On the contrary, cold rolling involves stretching and rolling the steel sheets at room temperatures. In steel rolling mills, especially in the hot rolling mills, the sheet/strip is rolled at variable speeds (4–15 m/s), since strip speed is inversely proportional to strip thickness. The width of the strip is approximately 2–2.5 m that require multiple cameras to capture the total width. If cameras are synchronized to millisecond levels of accuracy, then either the defects would be missed or the location would be misidentified. Therefore, the proposed algorithm and framework providing microsecond-level synchronization of cameras is crucial for such high-speed machine-vision applications.

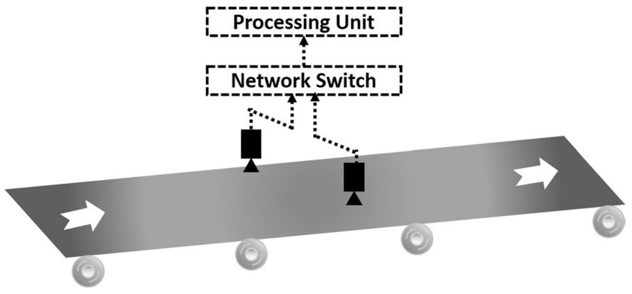


Fig. 1 Illustration of the machine-vision-based steel surface inspection

3 Related and relevant work

The crux of the proposed system is the temporal synchronization and in this section, the synchronization methodologies, and its classification into hardware, software, and network-based synchronization is discussed.

3.1 Hardware-based synchronization (HBS)

HBS entails synchronizing cameras using external circuits. Different variants have been prevalent in the entertainment industry since the time of analog cameras and display devices. One methodology involves transmitting time-codes of the Society of Motion Picture and Television Engineers (SMPTE) through wireless networks such as WiFi [12]. In another variant, a synchronization signal is transmitted over firewire to the camera array and redistributed through ethernet cables [13]. However, the synchronization performance depends on the operating system of the computer used to connect the cameras [14]. Another attempt proposes an Arduino-based triggering circuit for synchronizing the camera with a structured light projector [15]. However, these HBS methodologies cannot be adopted easily in industrial environments as additional cabling is needed that is undesirable. Synchronization reliability depends on the reliability and ruggedness of external circuit to operate in harsh industrial environments.

3.2 Software-based synchronization (SBS)

Unlike HBS, where additional cabling and external circuits are required, SBS entails synchronizing the cameras based on numerous features extracted from the acquired images or video. One such attempt synchronizes videos taken from multiple devices/cameras based on frame rate, camera flashes, or audio and video fingerprints [16]. Another attempt registers a methodology of video-alignment based on observations of a set of objects to model the alignment as frame-association rather than traditional continuous time-warping [17]. Next, a methodology of synchronizing high-speed vision sensors is presented that uses an illumination-based technique derived from the phase-locked-loop (PLL) algorithm [18]. Even though SBS is easy to implement in an industrial environment, the methodologies do not provide repeatable results. Moreover, SBS synchronizes only frames in which the features matched, so not all frames are synchronized.

3.3 Network-based synchronization (NBS)

These limitations were addressed with the introduction of NBS. With the advent of digital cameras networked using UDP and TCPIP protocols, cameras were synchronized using the network-time-protocol (NTP) [19] for the synchronization of clocks between various computer systems through packet-switched, variable-latency data networks [20]. While NTP provides synchronization accuracy in the millisecond range, high-speed industrial applications require higher accuracy that led to the introduction of IEEE-1588 standard [21] called precision-time-protocol (PTP). For improving the accuracy, precision, and robustness of PTP, the revised standard was introduced in the year 2008 called IEEE-1588-2008, more commonly known as precision-time-protocol version 2 (PTP-V2). However, this revision was not backward compatible with the first version making it very difficult to use in practical implementations. To ease practical usage of PTP, the IEEE-1588-2019, informally known as PTPv2.1, was introduced that included backward-compatible improvements. This gave rise to control applications that use PTP for clock synchronization [22–24].

Even though several publications mentioning the synchronization of cameras using PTP are available, in reality, these are attempts at synchronizing computers connected to a network using PTP, while the cameras are still synchronized using NTP with the computers [25]. Hence, the synchronization procedure involves synchronization of multiple computers connected in a network, rather than the cameras that is essentially the same as NTP synchronization between the cameras [26]. Another method of synchronizing multiple high frame-rate cameras using PTP describes a master–slave architecture, where each camera connects to the computer through a frame grabber card. The clocks of the master computers in the architecture are synchronized using PTP [27]. There have been many similar attempts of such synchronization of computers [28–31] that are connected to devices [32, 33]. These papers claim that the computers are synchronized in the microsecond range, but do not consider the transmission delays between the MV cameras and computers. The overall synchronization times of MV cameras increase to millisecond range.

The transmission delays between the MV cameras and computers could not be estimated, since there were no clocks inherently in the cameras. For the accurate estimation of the time differences, it is essential that the clocks of sensors and computers are separate so that the transmission and reception time of the data from the sensors to the computing units can be estimated for accurate synchronization. This limitation can be overcome by directly synchronizing device clocks rather than synchronizing the clocks of computers that are connected to devices. Accomplishing this requires that the devices have internal clocks that can be synchronized. PTPv2.1 gave rise to the development of cameras with internal clocks that can be synchronized. However, PTP was not used to synchronize MV cameras, since PTPv2.1 is a network network-intensive protocol that frequently sends numerous messages to all cameras and the cameras also transmit high-resolution images, thereby causing network congestion.

To understand the need for another synchronization algorithm, we would like to present the disadvantages of existing synchronization methodologies, NTP and PTP. The drawbacks of NTP synchronization methodology can be given as follows:

- NTP was primarily designed for synchronization over the Internet using unicast methodology for synchronization primarily but can also be used in multimode metholodology. Although NTP can keep the system clock accurate, the clock itself has a very low resolution. It updates the clock value, or ticks, once every 10–15 ms and thereby the synchronization accuracies are in the millisecond range.
- NTP is an unidirectional synchronization methodology where the timestamp is sent from the master devices to its client devices, but the feedback correction from the clients is not considered. Without the feedback correction, errors in the synchronization between the client devices are introduced which, in turn, affect the synchronization accuracy.
- In NTP, the selection of clock source or master happens on the client side. Generally, an NTP client receives

timestamps from all of its possible masters in the network and then client decides which master clock are acceptable for synchronization. Since the master keeps changing over a period of time, there are chances of errors being introduced in the network, thereby reducing the synchronization accuracies.

Similarly, the drawbacks of PTP synchronization methodology can be given as follows:

- PTP was designed for local networks with multicast transmission and, in ideal conditions, the system clock can be synchronized with sub-microsecond accuracy to the reference time. Multicast transmission involves synchronizing all the client devices simultaneously. Even though the PTP has correction mechanisms inbuilt in the protocol, the underlying condition for its accuracy is the network congestion to be minimal, so that maximum bandwidth can be utilized for the PTP protocol. In a congested network where bandwidth is limited, simultaneously sending multiple transmissions for synchronization to numerous sensors that causes delays in the network and thereby affects synchronization accuracy.
- In PTP there are slaves, which are synchronized to their masters. Each communication path has one master and its slaves can be masters on other communication paths. The synchronization is carried out using best master clock (BMC) algorithm which would enable the master clock to be changed as per the shortest path. Hence, there is no way to ensure that there would be a single master in the network and thereby introduce errors in the synchronization times.
- PTP is network-intensive that does not allow for transmission of network-intensive data such as images that would cause network choking. This would render the synchronization process unusable for high bandwidth data transmission applications.

This paper aims to solve this problem by proposing a novel synchronization algorithm and framework for multiple sensors and in this case, MV cameras, while ensuring bandwidth availability for image transmission.

4 Problem statement

Consider a L node sensor network comprising $(L-1)$ sensors and 1 computing unit, as shown in Fig. 2. The basic premise of the synchronization problem is that all nodes contain an internal clock/counter for building an estimate of global time reference, τ. The clock of each node has its own local state that can be given as [34]

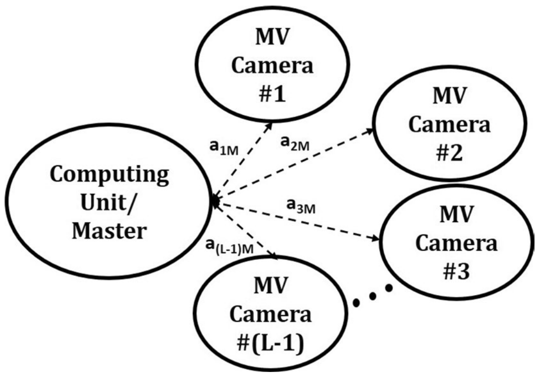


Fig. 2 Structure of MV camera nodes in a network

$$C_i(\tau) = \theta_i + \phi_i \cdot \tau + \eta_i \cdot \tau^2 + \xi_i, \tag{1}$$

where $C_i(\tau)$ denotes the local clock value; θ_i is the local clock offset, T_0; ϕ_i is the local clock skew; η_i is the clock drift caused by clock aging. Furthermore, ξ_i represents the clock jitter, that is modeled as zero-mean white Gaussian noise with standard deviation σ_L, where L is the number of clock nodes.

In solving this synchronization problem, for convenience, a simplified model is considered where the quadratic term in Eq. (1), that is often referred to as clock aging, is neglected. Also, the clock jitter ξ_i has little influence on the clock value. Therefore, the clock value at each node can then be given as

$$C_i(\tau) = \theta_i + \phi_i \cdot \tau. \tag{2}$$

The problem of clock synchronization is basically reduced to the estimation of θ_i and ϕ_i, so that the clocks of the nodes can be adjusted. But usually, the global time reference τ is not available at each node, making it impossible to compute θ_i and ϕ_i. Therefore, it can be inferred that the computing unit and MV cameras cannot be accurately synchronized with each other. Due to this limitation of temporal synchronization with τ, MV cameras cannot be synchronized using this technique where synchronous capture of images is the main objective.

To overcome this limitation, the parameters θ_i and ϕ_i can be obtained indirectly by measuring the clock of each node with respect to another node in the network. In this MV camera synchronization problem, this can be accomplished by synchronizing the $(L-1)$ MV cameras with the 1 computing unit. Therefore, if we solve Eq. (2) for τ and substitute into the equation of the computing unit, then the clock value of the computing unit denoted as master, M, can be given as

$$C_M = \theta_{iM} + \phi_{iM} \cdot C_i. \tag{3}$$

In fact, the computing unit or Master to which $(L-1)$ MV cameras are synchronized can be given as

$$C_M(\tau) = \theta_M + \phi_M \cdot \tau. \tag{4}$$

For solving the synchronization problem, the exact values of (θ_M, ϕ_M) are not important; rather, it is critical that all MV camera nodes converge to one common master reference node. Also, the parameters (θ_M, ϕ_M) can be calculated as the computing unit is the only node in the network in which the global time reference is available. Every MV camera node keeps an estimate of the master reference time from Eq. (4) using a linear function described as

$$\hat{C}_i = \hat{o}_i + \hat{s}_i \cdot C_i. \tag{5}$$

The goal is to determine $(\hat{o}_i, \hat{s}_i)$ for every node in network, such that

$$\lim_{\tau\to\infty} C_i\hat{(}\tau) = C_M(\tau) \text{ for } i, k = 1, 2, \ldots, L. \tag{6}$$

Thereby, the synchronization problem is formulated based on Eq. (6), where all nodes need to be synchronized with a common master reference node or computing unit.

5 Proposed algorithm for synchronizing two sensors

To solve the problem stated in the previous section and overcome the drawbacks of the existing synchronization methodologies mentioned above, a temporal synchronization algorithm is proposed. In the proposed algorithm, the sensors are synchronized with the computing unit instead of global time reference. The computing unit acts as the master and MV cameras act as slaves. The important aspect to understand is that each sensor synchronizes separately with the computing unit. Consider L nodes in the network comprising $(L-1)$ sensors and 1 computing unit. Even though there is physically only one computing unit connected to sensors through network switches, as per the master–slave architecture, each sensor is connected to the computing unit individually, leading to $(L-1)$ master–slave pairs, as shown in Fig. 3.

In the proposed algorithm, synchronization messages with timestamps are transacted between the master (processing unit) and slave clock (sensor) through the transparent clocks (network switch) acting as bridges. Consider the timing diagram shown in Fig. 4, where τ_m is the time at the master. τ_1 is the transmission time of the synchronization message by the master, and τ_2 is the reception time of the synchronization message sent through the transparent clocks.

The delay between the receiving and forwarding of the signal at the transparent clock is termed as the residence

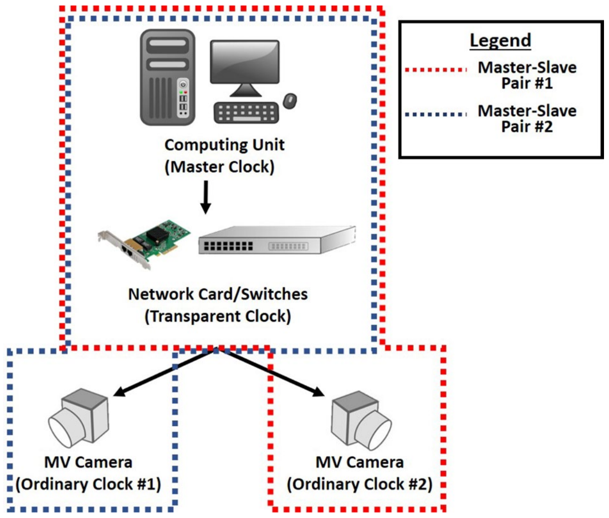

Legend
Master-Slave Pair #1
Master-Slave Pair #2
Computing Unit (Master Clock)
Network Card/Switches (Transparent Clock)
MV Camera (Ordinary Clock #1)
MV Camera (Ordinary Clock #2)

Fig. 3 Master–slave hierarchical architecture

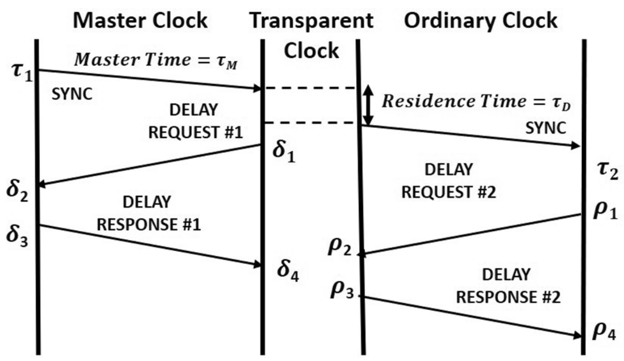


Fig. 4 Timing diagram of proposed algorithm

time, τ_d. This is the delay introduced into the network by the addition of transparent clock. Therefore, the time of synchronization at the slave, τ_s, can be given as

$$\tau_s = \tau_m + \Delta, \tag{7}$$

where Δ is the offset error between the transmission and reception of the synchronization signal which can be given as

$$\Delta = \tau_2 - \tau_1 - \lambda_2 - \mu. \tag{8}$$

In Eq. (8), τ_1 and τ_2 are known timestamps of the master and slave, respectively. λ_2 is the mean delay of the path between the slave and the transparent clocks, and μ is the correction field applied between the master and the transparent clocks.

As shown in Fig. 4, once the synchronization signal is received by the slave, a delay request signal is sent from the slave at time ρ_1 to the transparent clock at time ρ_2. Then, slave receives the delay response at time ρ_4 from the transparent clock at time ρ_3. Similarly, a delay request signal is sent from the transparent clock at time δ_1 to the master clock at time δ_2. Then, the transparent clock receives the delay response at time δ_4 from the master clock at time δ_3. Furthermore, the mean delay of the path can be defined as the average of the time taken to send the delay request and obtain the delay response. Hence, the mean path delay between the slave and transparent clocks, λ_2, can be mathematically given as

$$\lambda_2 = \frac{((\rho_2 - \rho_1) + (\rho_4 - \rho_3))}{2}. \tag{9}$$

$$\Delta = \tau_2 - \tau_1 - \frac{((\rho_2 - \rho_1) + (\rho_4 - \rho_3))}{2} - \frac{((\delta_2 - \delta_1) + (\delta_4 - \delta_3))}{2} + \tau_d. \tag{13}$$

Now that, all of the above terms in the right are measured, Δ can easily be computed.

Finally, substituting Eq. (13) in Eq. (7), time of synchronization at the slave, τ_s can be given as

$$\tau_s = \tau_m + \tau_2 - \tau_1 - \frac{((\rho_2 - \rho_1) + (\rho_4 - \rho_3))}{2} - \frac{((\delta_2 - \delta_1) + (\delta_4 - \delta_3))}{2} + \tau_d. \tag{14}$$

Algorithm 1 Algorithm of the proposed approach to synchronize two sensors

```
    Input : $\tau_1, \tau_2, \rho_1, \rho_2, \rho_3, \rho_4, \delta_1, \delta_2, \delta_3, \delta_4,$
 1: while $\Delta \not\approx 0$ do
 2:     Sync message ($\tau_M$) is sent from master
 3:     Compute the mean path delays $\lambda_1$ and $\lambda_2$
 4:     Assess the correction field $\mu$
 5:     Determine offset error $\Delta$
 6:     Calculate corrections $\theta_{adj}$ and $\phi_{adj}$
 7:     Finally, Compute Slave time $\tau_s$
Ensure: $\tau_M \approx \tau_s$
```

Similarly, the mean path delay between the transparent and master clocks, λ_1, can be mathematically given as

$$\lambda_1 = \frac{((\delta_2 - \delta_1) + (\delta_4 - \delta_3))}{2}. \tag{10}$$

The correction field applied between the master and the transparent clocks, μ, can be given as function of residence time and the mean path delay between the transparent and master clocks as follows:

$$\mu = \lambda_1 + \tau_d. \tag{11}$$

Substituting Eq. (10) in Eq. (11), the correction field, μ, can be further given as

$$\mu = \frac{((\delta_2 - \delta_1) + (\delta_4 - \delta_3))}{2} + \tau_d. \tag{12}$$

Furthermore, substituting Eqs. (12) and (9) in Eq. (8)

From Eq. (14), time of synchronization at the slave, τ_s can be calculated accurately, since all the other parameters are deterministic and known values as shown in Algorithm 1. To further improve the synchronization accuracy, the proposed algorithm comprises slave clock correction. This correction consists of both offset and skew adjustment, denoted by θ_{adj} and ϕ_{adj}, is given as

$$\theta_{\text{adj}} = -\zeta \cdot \theta_{\text{est}}(n)\phi_{\text{adj}} = -\eta \cdot \phi_{\text{est}}(n), \tag{15}$$

where $\theta_{\text{est}}(n)$ and $\phi_{\text{est}}(n)$ are the estimated offset and skew values; ζ and η are the attenuation coefficients having values between (0, 1). The slave clocks are corrected by a fraction of the offset and skew estimates. This ensures over-correction is avoided. However, the tradeoff is that it takes longer to synchronize the sensors. Therefore, offset and delays are computed periodically to improve the synchronization accuracy provided the jitters in the measurements are negligible.

To understand the novelty of proposed algorithm, the salient differences with PTP are given below:

- PTP for multiple devices/sensors works with multicast mode of operation by which synchronization signals are sent simultaneously to all the sensors. This works well in networks that are not congested. The proposed algorithm treats each of the sensors separately and operates in unicast mode. This enables the proposed algorithm to synchronize effectively in congested networks also.
- For effective correction of errors such as skew and drift of the clocks, PTP repeatedly sends synchronization signals and corrects the clocks. In the proposed algorithm, due to the introduction of slave clock correction factor, the synchronization is accurate and corrects itself periodically.

6 Proposed framework for synchronizing a network of sensors

Next, a framework is proposed that is based on the synchronization algorithm introduced in the previous section. The basic objective of the proposed framework is to synchronize and seamlessly integrate multiple MV cameras. The salient feature is that sensors can be easily added or removed. This framework comprises multiple master–slave network segments. Each of this segments are synchronized individually at different time instants. Consider $(L-1)$ MV sensors connected to a computing unit, and then, Eq. (14) for $(L-1)$ network segments can be given as

$$\begin{aligned}\tau_{si} = \tau_{mi} + \tau_{2i} - \tau_{1i} - \frac{((\rho_{2i} - \rho_{1i}) + (\rho_{4i} - \rho_{3i}))}{2} \\ - \frac{((\delta_{2i} - \delta_{1i}) + (\delta_{4i} - \delta_{3i}))}{2} \\ + \tau_{di} \text{ for } i = 1, 2, \ldots, (L-1).\end{aligned} \tag{16}$$

Physically, computing unit is the same for all network segments, since there is only one computing unit connected to all the sensors

$$\tau_{mi} = \tau_{mk} = \tau_m \text{ for } i, k = 1, 2, \ldots, (L-1). \tag{17}$$

Substituting Eq. (17) in Eq. (16)

$$\begin{aligned}\tau_{si} = \tau_m + \tau_{2i} - \tau_{1i} - \frac{((\rho_{2i} - \rho_{1i}) + (\rho_{4i} - \rho_{3i}))}{2} \\ - \frac{((\delta_{2i} - \delta_{1i}) + (\delta_{4i} - \delta_{3i}))}{2} \\ + \tau_{di} \text{ for } i = 1, 2, \ldots, (L-1).\end{aligned} \tag{18}$$

This implies that each of the sensors is separately trying to synchronize with the master. Although Eq. (18) shows that τ_{si} is calculated with high accuracy from the first frame, in practice, this is an iterative process where after the first few cycles, the synchronization time τ_{si} remains stable and constant in each segment. Thus, this synchronization process is carried out periodically, so that the accumulation of errors is avoided without causing network congestion. Periodic synchronization is accomplished using a UNICAST mode of transmission that sends the synchronization messages to only one slave at a time. To ensure limited network bandwidth utilization, only one sensor is to be synchronized every α seconds, which is determined based on the number of sensors in the network. The time-period of minimum synchronization for the $(L-1)$ sensors in the framework can be calculated as

$$\Omega = (L-1) \times \alpha \text{ s}. \tag{19}$$

Algorithm 2 Algorithm of the proposed framework to synchronize a network of sensors

```
    Input : $(\tau_{1i}, \tau_{2i}, \rho_{1i}, \rho_{2i}, \rho_{3i}, \rho_{4i}, \delta_{1i}, \delta_{2i}, \delta_{3i}, \delta_{4i}$ for all slaves
    for each Slave $s_i \in \mathcal{S}$ do
2:      while $\Delta_i \not\approx 0$ do
            Sync message $(\tau_M)$ is sent from master
4:          Compute the mean path delays $\lambda_{1i}$ and $\lambda_{2i}$
            Evaluate the correction field $\mu_i$
6:          Assess offset error $\Delta_i$
            Calculate adjustments $\theta_{adj_i}$ and $\phi_{adj_i}$
8:          Finally, Compute Slave time $\tau_{si}$
        Wait every $\alpha$ seconds
Ensure: $\tau_M \approx \tau_{si}$ for all slaves $s_i \in \mathcal{S}$
```

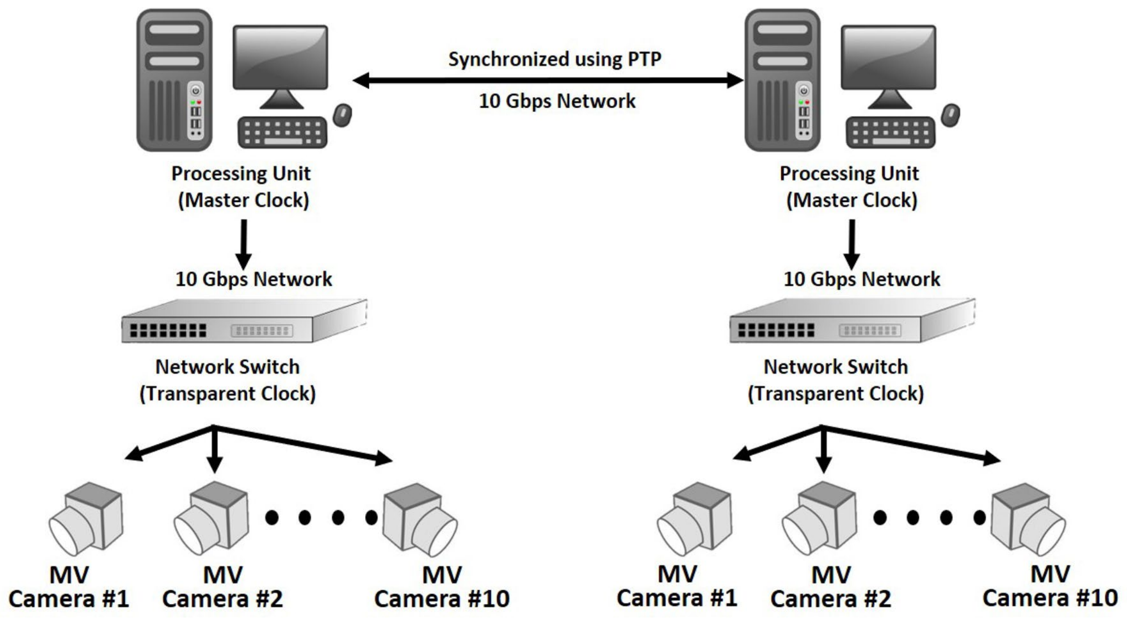
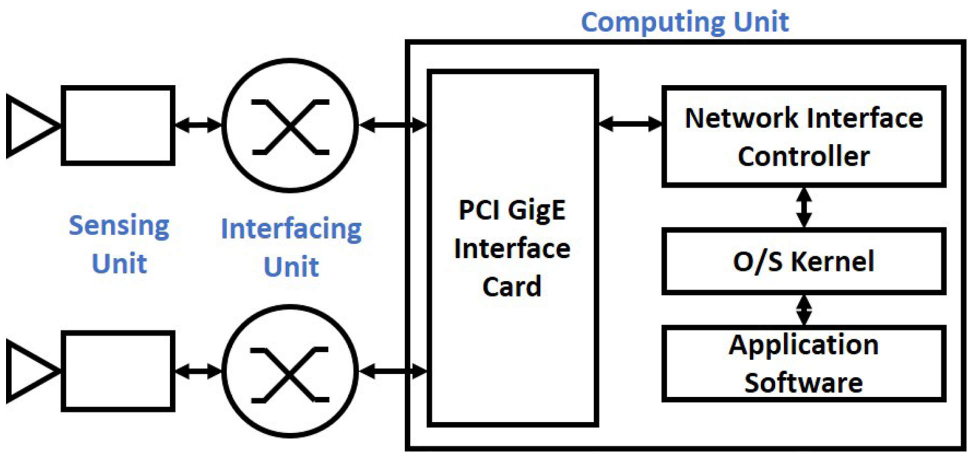


Fig. 5 Illustration of the proposed framework with 20 MV cameras

Fig. 6 Schematic of hardware configuration



The algorithm for synchronizing the framework is shown in Algorithm 2. Even though theoretically, any number of sensors can be connected in such a manner, the number of sensors connected to a computing unit is constrained by the network bandwidth, because sensors need to transfer the image buffers through the network, and typically, the requirement is 1 Gigabit per second (Gbps) per sensor. For example, consider a computing unit and network switch in a 10 Gbps network. A maximum of 10 sensors can be connected to this network. If more sensors are to be connected, another computing unit with a similar 10 Gbps network is added, as shown in Fig. 5. While the sensors are synchronized with computing unit using the proposed algorithm, multiple computing units are synchronized with each other using PTP.

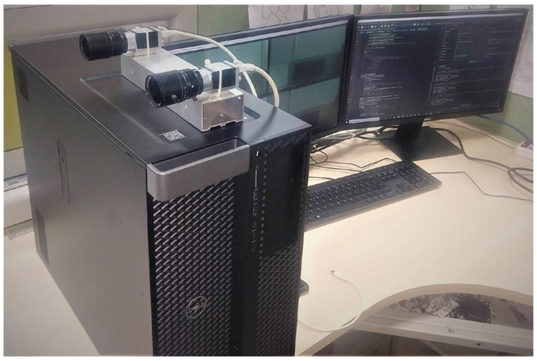


Fig. 7 Final hardware setup for the proposed system

7 Experimental setup

7.1 Hardware configuration

The hardware configuration of the proposed setup, as shown in Fig. 6, primarily consists of three major components—sensing, interfacing, and processing units.

The sensing units of the proposed system comprise 1/2" CMOS-based color area-scan cameras (Basler ACE acA1300-75gc) coupled with a C-Mount lens (Goyo - GM38013MCN-1) with focal length of 8 mm and minimum object distance of 200 mm. The cameras are also PTP compliant that implies the presence of internal clock. The interfacing unit consists of a peripheral component interconnect (PCI) based four-port ethernet card (Adlink PCIe-GIE64+), power-over-ethernet (POE) feature, and also PTP compliant. The processing unit consists of a computer powered using an i7 processor, 16 Gigabytes of RAM, and 4-gigabyte Nvidia graphics card. The operating system used is Windows, because drivers for Basler cameras are more stable in the Windows environment. The final hardware configuration and setup is shown in Fig. 7.

7.2 Application development

Linux operating system offers the flexibility for accurate recording of timestamps and monitoring the network, while the WINDOWS operating system offers stability for the drivers of the cameras. Thus, to overcome this predicament, the proposed synchronization methodology was developed in C and executed in the WINDOWS subsystem for LINUX [35] environment. This subsystem operates as the grandmaster clock for the proposed approach. The network synchronization was monitored using a network monitoring tool such as WIRESHARK [36]. This tool helps in visualizing the various messages and timestamps transmitted and received in the network. The messages and timestamp comprises:

- The type of message such as announce, sync, or delay request is provided.
- Source and destination Internet protocol (IP) addresses of the messages for identification are provided.
- The length of the message transmitted/received is provided.
- The timestamp is provided in microseconds as a time difference between the reception time at the destination and the transmission time at the source.

The image and video capture applications need to use the drivers of cameras, and hence, these applications were developed in the Python environment, an interpreted, high-level, general-purpose programming language. A flowchart in Fig. 8 shows the interaction of the various environments and applications developed.

8 Experimental results and discussion

The evaluation of the proposed approach was carried out using the experimental setup depicted in Sect. 7. The algorithm uses the proposed approach to synchronize multiple sensors, in this case, cameras. Following the synchronization, images were grabbed from the cameras and the time of image grabbing from the two cameras were recorded. The absolute difference between the acquisition times of cameras (denoted as synchronization time) is computed. The above steps were repeated continuously at a rate of 75 frames/s upto a period of 2 days to check for deviation. The results of these experiments using proposed approach are presented in this section.

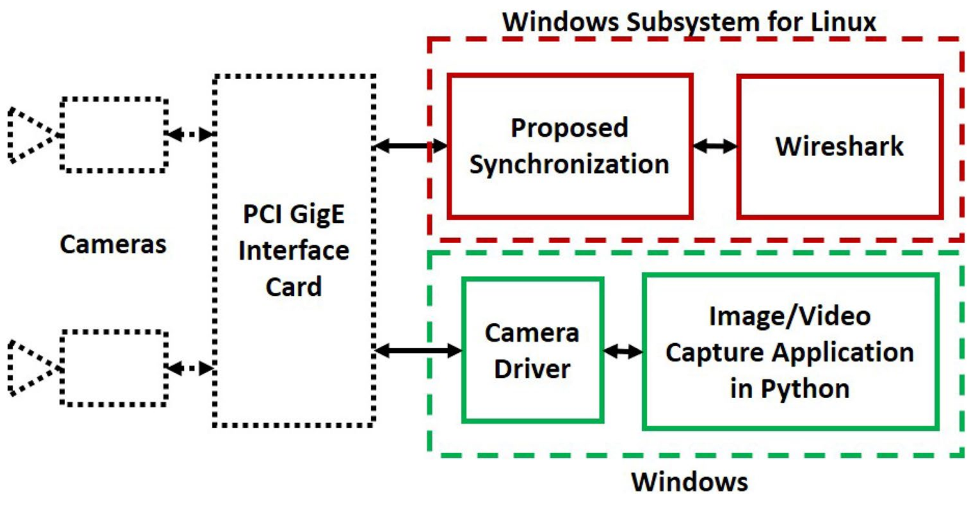


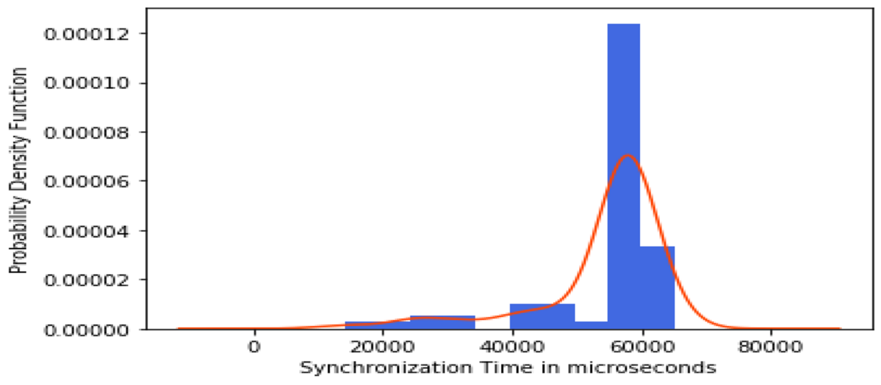
Fig. 8 Flowchart depicting the interaction of various environments and applications

Table 1 Comparison of mean, median, and standard deviation of synchronization times between existing and proposed approaches with image capture

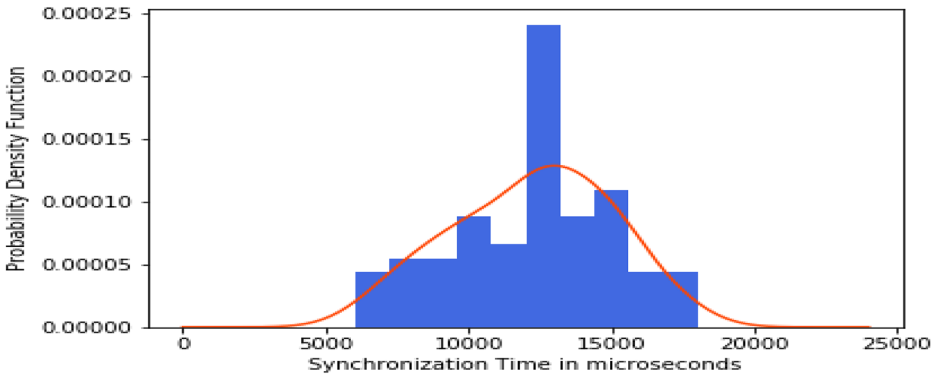
	Synchronization time using NTP (ms)	Synchronization time using PTP (ms)	Proposed synchronization time (ms)
Mean	54197.37	12223.68	6.87
Median	57000	12500	7.00
St. Dev.	9742.77	2760.62	1.58

To ensure the precision and accuracy of the proposed approach, a statistical analysis of the results is conducted. Furthermore, the results of synchronization times of two cameras in 2, 3, and 4 camera networks are analyzed . In each of these different experiments, the cameras are synchronized and acquired at the maximum frame rate of 75 frames a second. These images were also transmitted from the cameras and received at the computing unit, thereby ensuring that the proposed approach uses optimal network bandwidth and does not cause network congestion.

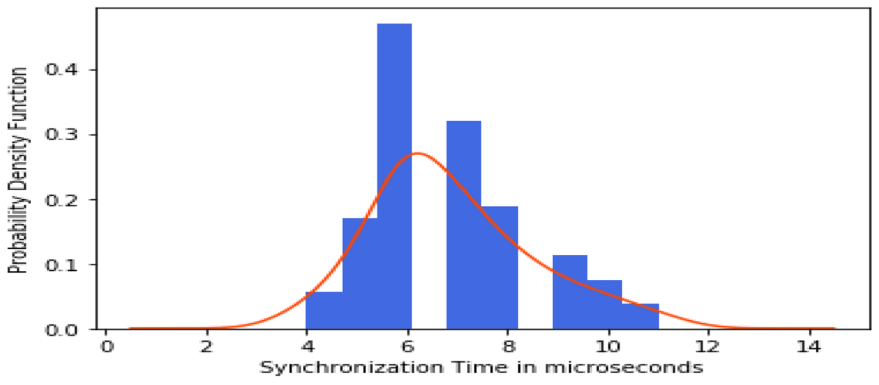
Fig. 9 Histogram and density plots of synchronization times between existing and proposed approaches



(a) Existing synchronization time using NTP in microseconds



(b) Existing synchronization time of computers connected to cameras using PTP in microseconds

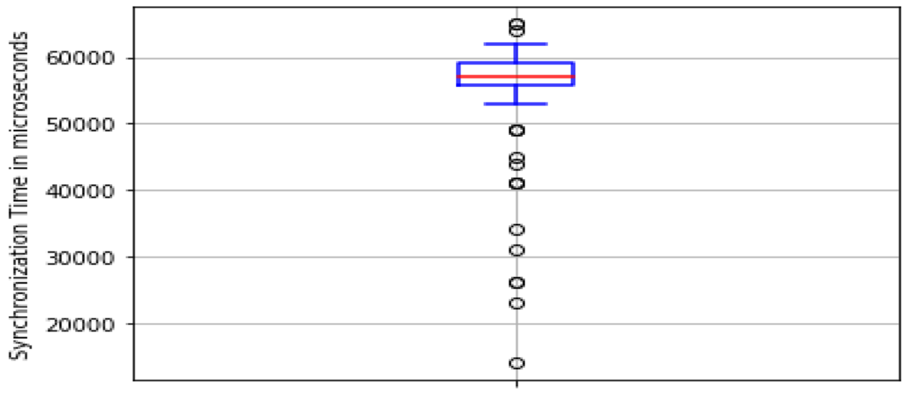


(c) Proposed Synchronization time in microseconds

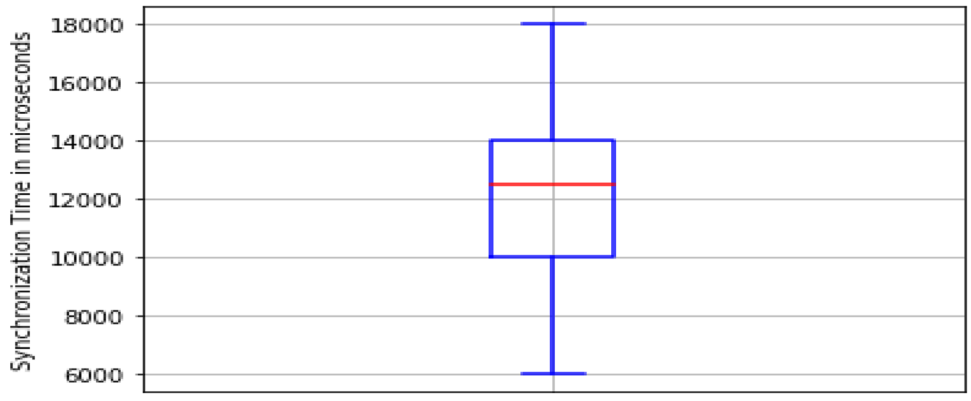
8.1 Results and comparison of the proposed approach with existing methodologies

The proposed approach is compared against two currently used state-of-art methodologies discussed in Sect. 3. In the first method, camera synchronization uses NTP in which all the nodes are synchronized with GPS, because all the master clocks in the network needs to be accurate. In the second method, cameras connect to computers that synchronize using PTP where the computers are synchronized with GPS rather than local workstation, because BMC algorithm could use any computer as master based on shortest path. Methodologies involving separate electronic circuits for synchronization, as discussed in Sect. 3, are not considered for this comparison, because long-distance cabling is not possible for industrial applications. Mean, median, and standard deviation of synchronization times (in milliseconds) using the existing methods, and the proposed approach are compared in Table 1. Therefore, the inference is that currently available methodologies of synchronization are not suitable

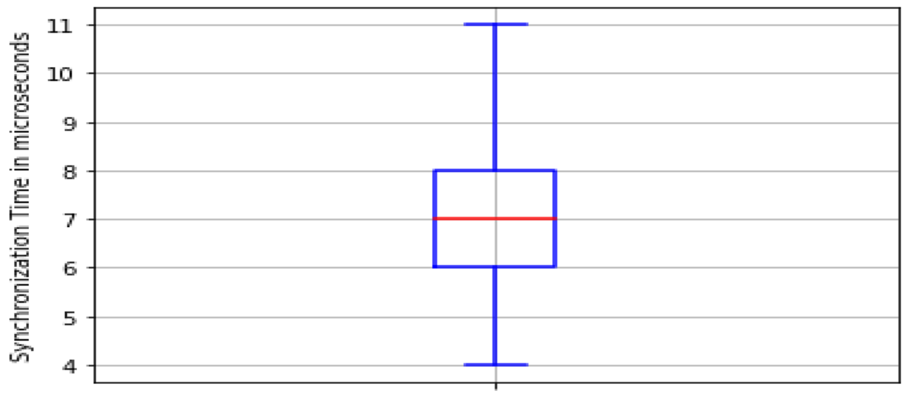
Fig. 10 Box-and-Whisker plots between existing and proposed approaches



(a) Existing synchronization methodology using NTP in microseconds



(b) Existing synchronization methodology of computers connected to cameras using PTP in microseconds



(c) Proposed Synchronization Approach in microseconds for MV cameras as synchronization requires microsecond-level accuracies are required. Moreover, the standard deviation value of 1.58 ms for the proposed approach implies that the proposed approach is superior to existing methodologies whose standard deviation values are 9743 ms and 2761 ms.

Figure 9 shows the histogram plots and Gaussian probability density function (pdf) of synchronization times for existing methodologies as well as the proposed approach. On analysis, the skewness factor for synchronization time using NTP is −2.3, while the skewness factor for synchronization time of computers connected to cameras using PTP is −0.19. This shows that synchronization using NTP has more weight in the right tail of the distribution. The skew factor of the proposed approach is 0.64 which shows that the weight of the distribution is in the left tail of the distribution. This implies that the synchronization times of existing methodologies are predominantly higher than the mean, while the synchronization time of the proposed approach is predominantly lower than the mean.

The box-and-Whisker (BW) plots shown in Fig. 10 depict the comparison of difference between capture times of the two cameras synchronized the existing methods and the proposed approach. The BW plot of synchronization methodology using NTP indicates that the spread between the upper and lower quartile is 4000 ms. But most importantly, the numerous outliers are indicated as circles in the plot, which imply that the degree of variation in synchronization times outside the lower and upper quartile is higher, making this synchronization methodology unpredictable and unsuitable for MV cameras. The BW plot of synchronization of computers using PTP that are connected to MV cameras shows that while there are no outliers, in this case, the spread between the upper and lower quartile is 2000 ms which are still significantly high, thereby making this synchronization methodology also unsuitable for MV cameras where microsecond-level accuracies are required. The BW plot of the proposed approach indicates that the spread between the upper and lower quartile is 2 ms without any outliers, thereby making this most suitable for synchronizing MV cameras.

In an another experiment, the sensors were synchronized with the master in a network without any image capture.

Table 2 Comparison of mean, median, and standard deviation of synchronization times between existing and proposed approaches without image capture

	Synchronization time using NTP (ms)	Synchronization time using PTP (ms)	Proposed synchronization time (ms)
Mean	46197.37	6.12	6.28
Median	47000	7.00	7.00
St. Dev.	6754.43	1.82	1.37

Table 3 Comparison of mean, median, and standard deviation of difference between capture times of the multiple cameras synchronized using the proposed approach

	Proposed framework in 2-camera network (ms)	Proposed framework in 3-camera network (ms)	Proposed framework in 4-camera network (ms)
Mean	6.87	7.51	7.89
Median	7.00	7.00	7.00
St. Dev.	1.59	1.77	1.84

This would mean that the sensors are synchronized in a network without congestion and the total bandwidth is available for synchronization. The comparison of the results of this experiment is provided in Table 2. The results show that the mean of PTP is superior to the proposed approach, but the standard deviation of the proposed approach is slightly lesser than PTP. These results indicate that in the absence of network congestion, both PTP and the proposed approach perform equally and either can be used in the absence of network congestion due to the high data-rate from the sensors.

8.2 Results of proposed framework

The experimental setup for the proposed framework extended the previous setup of cameras to 2-, 3-, and 4-camera networks. The variation of synchronization times between the first and second cameras in a 2-, 3-, and 4-camera network is tabulated in Table 3. The box-and-Whisker plot of the comparison between the 2-, 3-, and 4-camera network is shown in Fig. 11. While the median values are the same in all camera networks, there is a small increase in both mean and standard deviation as the number of cameras in the network increases. Also, as the number of cameras in the network increases, the upper quartile is larger than the lower quartile of the BW plots. The reason for this variation is that after each run of synchronization, images are transported

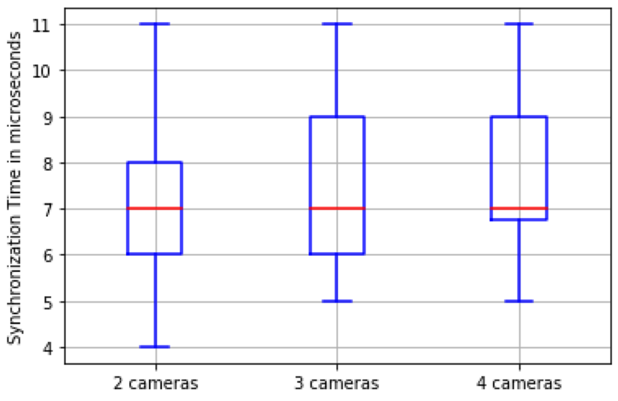
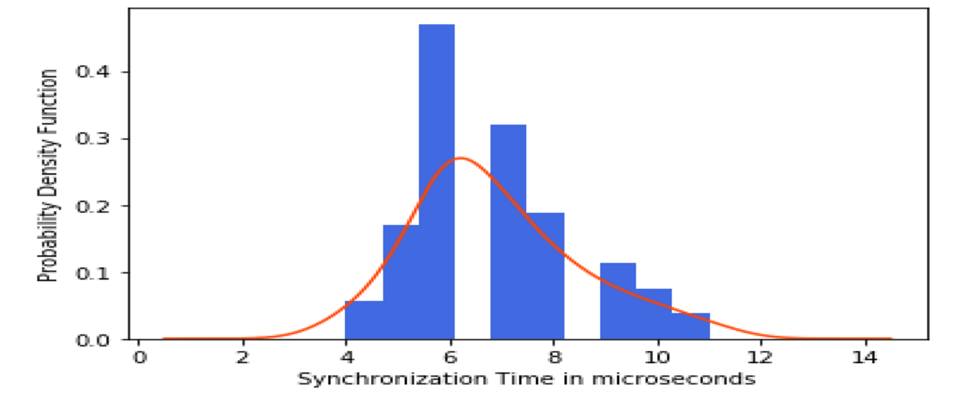
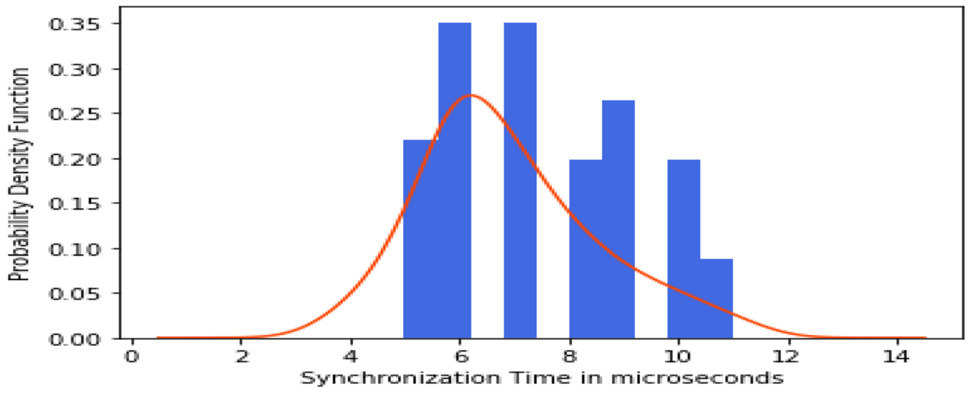


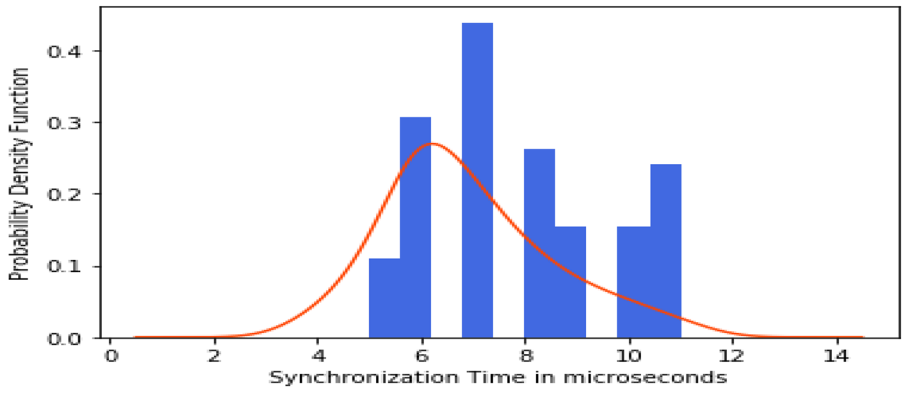
Fig. 11 Box-and-Whisker plots of synchronization times between 2 cameras in a 2-camera, 3-camera, and 4-camera network



(a) Histogram and PDF of synchronization times in 2 Camera Network



(b) Histogram and PDF of synchronization times in 3 Camera Network



(c) Histogram and PDF of synchronization times in 4 Camera Network

Fig. 12 Histogram and PDF plots of difference between capture times between 2 cameras in a 2-camera, 3-camera, and 4-camera network synchronized using the proposed approach

from the cameras to the computing unit, thereby increasing the load in the network bandwidth. However, this increase in the mean and standard deviation is insignificant, because the absolute values of increase in the mean are about 0.8 ms and the standard deviation is 0.08 ms.

Figure 12 shows the histogram and PDF for the synchronization times between 2 cameras in 2-, 3-, and 4-camera networks. While the plot for the 2-camera network depicts unimodal Gaussian distribution, the plot for 3- and 4-camera network significantly moves toward multimodal Gaussian distribution. The two peaks are significantly separated in the 4-camera network and is expected to increase as more cameras are added into the network. This implies that as more cameras are added in to the network, the modes are distinctly separated, thereby creating local maxima of synchronization times and limiting the maximum number of cameras that can be added to the network.

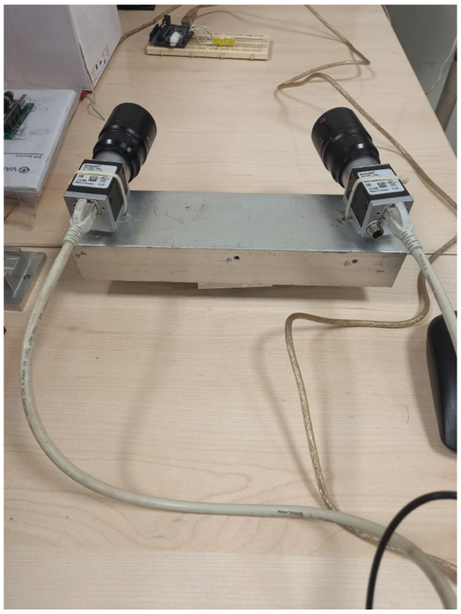


Fig. 13 Experiment setup with cameras viewing the Arduino-based LED circuit

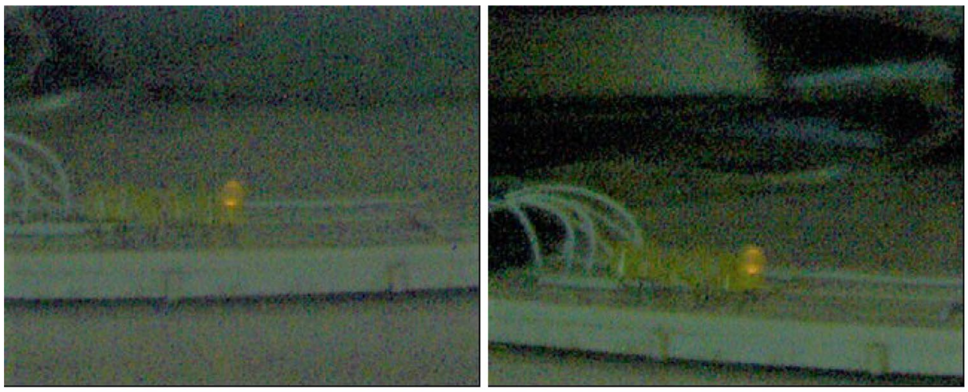
9 Results verification for image capture

The verification of results obtained from the proposed approach is accomplished by visually reviewing the images acquired from the cameras. As part of this experiment, an arduino-based circuit that has a sequence of LEDs that blink with a resolution of 8 microseconds. These LEDs are placed in the FOV of cameras and this setup is as shown in Fig. 13. The underlying assumption of this experiment is that if the cameras have been synchronized with microsecond levels of accuracy, then the images captured by the cameras of the LEDs should show the same LEDs lit. Based on this assumption, the experiments were conducted. One such set of images is shown in Fig. 14. The results show that the two cameras (distinguished by the angle of the object) have been synchronized to microsecond levels of accuracy, thereby verifying the results obtained in the previous subsection.

9.1 Results verification for video capture

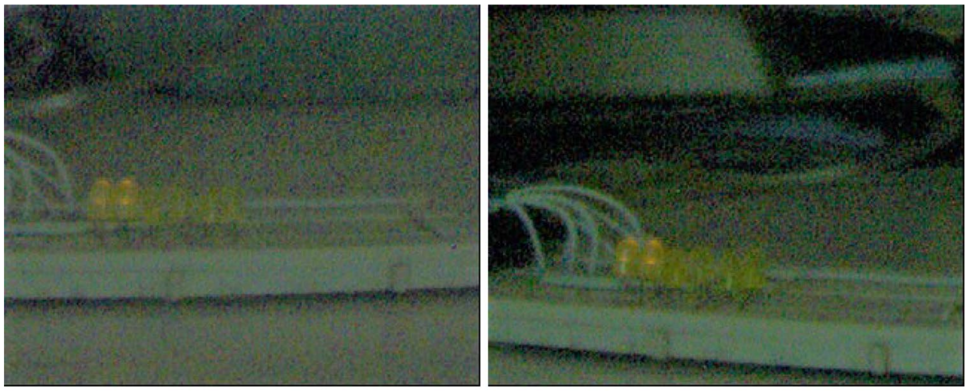
To review the performance of the proposed approach for video capture, the cameras were synchronized by the

Fig. 14 Images of Arduino-based LED circuit captured with cameras synchronized using the proposed approach



(a) Left Camera (b) Right Camera

Fig. 15 Comparison of frames from video capture using cameras synchronized using the proposed approach



(a) Left Camera (b) Right Camera

Table 4 Comparison of mean, median, standard deviation, and variance of difference between capture times of the two cameras synchronized using the proposed approach for multiple trials in microseconds

	Mean (ms)	Median (ms)	Standard deviation (ms)	Variance (ms)
Trial 1	7.00	7.00	1.94	3.77
Trial 2	7.08	7.00	1.70	2.90
Trial 3	7.64	7.00	2.23	4.98
Trial 4	6.92	6.00	1.82	3.31
Trial 5	6.68	6.00	2.15	4.63
Trial 6	7.09	7.00	1.74	3.04
Trial 7	7.47	7.00	1.95	3.80
Trial 8	7.63	7.00	2.16	4.68
Trial 9	6.99	7.00	1.77	3.14
Trial 10	7.32	7.00	2.16	4.69

Table 5 Comparison of network utilization and CPU utilization between existing and proposed approaches

	Synchronization and image capture using NTP (%)	Synchronization and image capture using PTP (%)	Proposed synchronization and image capture (%)
Network utilization	8	84	23
CPU utilization	26	32	36

proposed approach and videos were captured for a duration of 1 h. The Arduino-based LED circuit used in the previous subsection was again placed in the FOV for the video capture. For differentiating this experiment from the image capture experiment, two LEDs blink at every instant during video capture. Every 1000th frame was extracted from the videos captured from each of the cameras. The comparative analysis of these frame captured from each of the cameras confirmed that the cameras were synchronized as the images showed the same LEDs blinking at the same frame. One such example of the 10000th frame from both the cameras is shown in Fig. 15.

9.2 Performance of proposed approach

To ensure the stability of the proposed approach at various illumination conditions and ambient conditions, Monte Carlo trials are conducted where the synchronization of cameras was carried out in 10 different outdoor and indoor locations at different times of the day (for varying illumination conditions). At each location, 77 images were captured from each of the cameras. The synchronization time was calculated for each set of images, and then, statistical parameters such as mean and standard deviation were computed.

Table 4 shows the mean, median, standard deviation, and variance of the difference between capture times of the two cameras synchronized using the proposed approach for multiple trials. It can be seen that variation in mean and standard deviation is minimal between various trials and the proposed approach is stable over time and the multiple trials.

9.3 Performance dependencies of the proposed approach

This section discusses the performance dependencies of the proposed approach to the network and CPU utilization in the WINDOWS environment. Table 5 compares the network and CPU utilization of NTP, PTP, and the proposed approach. It is important to note that the network utilization includes both the usage for synchronization methodology as well as the transmission of the images from the camera. On analysis of this data, it is evident that network and CPU utilization is minimum. In case of NTP-based synchronization. In the case of PTP-based synchronization, while the CPU utilization is within acceptable limits, the network utilization is very high, thereby making it increasingly difficult for other devices to transmit and receive data in the network. For the proposed approach, while the network utilization is higher than NTP-based approaches, it is considerably lower than PTP-based approaches, thereby enabling its usage in robotic applications where other devices and functions would require network bandwidth for communication. Although, the CPU utilization is highest in the proposed approach, owing to the fact that the Windows subsystem for LINUX, it is still within acceptable limits and other applications can still be executed without any issues.

10 Conclusions

To conclude, a novel synchronization algorithm and framework of sensors, with microsecond-level accuracy is proposed. Moreover, the results and mathematical analysis of the proposed algorithm show that the synchronization accuracies required for industrial applications are achieved and the performance of this algorithm is far superior to the currently available technologies. With the understanding of the proposed algorithm and framework, this algorithm and framework can be extended to other fields and devices.

One of the main reasons for developing/proposing this algorithm and framework is its usage in the real-time image and video stitching applications where images from multiple cameras need to be captured at the same instant of time to ensure that the scene does not change, so that overlapping areas that are used for stitching do not change. The bandwidth of the network of cameras is congested due to the high bandwidth requirement for transmission and reception of images. Using the currently available NTP-based synchronization approach for real-time image stitching applications of moving objects, issues such as motion blur and unmatched scenes arise that deteriorate the results of stitching/melding the images or videos. This proposed synchronization algorithm and framework solves such issues and enables improvements in the field of image and video stitching.

Acknowledgements The authors would like to express their gratitude and thank the Automation Division of Tata Steel, Jamshedpur, Jharkhand, India for giving us the opportunity and allowing us to use their state-of-the-art laboratory facilities to conduct this research.

References

1. Horst, R., Negin, M.: Vision system for high-resolution dimensional measurements and on-line SPC: web process application. IEEE Trans. Ind. Appl. **28**, 993–997 (1992)
2. Jain, R., Kasturi, R., Schunck, B.G.: Machine vision. McGraw-Hill Inc, New York (1995)
3. Nayar, S.: Catadioptric omnidirectional camera. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition, pp. 482–488 (1997)
4. Lamkin, M., Ringgenberg, K., Lamkin, J.: Distributed multi-aperture camera array. US Patent US 2019 / 0246044 A1 (2019)
5. Keep one eye out (2012) [Online]. https://www.aviationtoday.com/2011/12/01/keep-one-eye-out/. Accessed 1 May 2020
6. Advanced distributed aperture system (adas) [Online]. https://www.ainonline.com/aviation-news/defense/2012-07-08/raytheon-adas-aids-helicopter-safety. Accessed 6 May 2020
7. Sanders-Reed, J., Koon, P.: Vision systems for manned and robotic ground vehicles. Proc. SPIE **7692**, 1–12 (2010)
8. Luo, Q., Fang, X., Liu, L., et al.: Automated visual defect detection for flat steel surface: a survey. IEEE Trans. Instrum. Meas. **69**(3), 626–644 (2020)
9. Sugimoto, T., Kawaguchi, T.: Development of a surface defect inspection system using radiant light from steel products in a hot rolling line. IEEE Trans. Instrum. Meas. **47**, 409–416 (1998)
10. Ghorai, S., Mukherjee, A., Gangadaran, M., et al.: Automatic defect detection on hot-rolled flat steel products. IEEE Trans. Instrum. Meas. **62**, 612–621 (2013)
11. Stojanovic, R., Mitropulos, P., Koulamas, C., et al.: Real-time vision-based system for textile fabric inspection. Real Time Imaging **7**, 507–518 (2001)
12. Meyer, F., Bahr, A., Lochmatter, T., et al.: Wireless GPS-based phase-locked synchronization system for outdoor environment. J. Biomech. **45**(1), 188–190 (2012)
13. Wilburn, B., Joshi, N., Vaish, V., et al.: High-speed videography using a dense camera array. Proc. IEEE CVPR **2**, 294–301 (2004)
14. Litos, G., Zabulis, X., Triantafyllidis, G.: Synchronous image acquisition based on network synchronization. In: Proceedings of CVPR workshops, p. 167 (2006)
15. Nguyen, H., Nguyen, D., Wang, Z., et al.: Real-time, high-accuracy 3D imaging and shape measurement. Appl. Opt. **54**(1), A9–A17 (2015)
16. Shrestha, P., Barbieri, M., Weda, H., et al.: Synchronization of multiple camera videos using audio-visual features. IEEE Trans. Multimed. **12**(1), 79–92 (2010)
17. Zini, L., Cavallaro, A., Odone, F.: Action-based multi-camera synchronization. IEEE J. Emerg. Sel. Top. Circuits Syst. **3**(2), 165–174 (2013)
18. Lei Hou, S., Hashimoto, K.: Illumination-based real-time contactless synchronization of high-speed vision sensors. In: 2008 IEEE international conference on robotics and biomimetics, pp. 1750–1755 (2009)
19. Litos, G., Zabulis, X., Triantafyllidis, G.: Synchronous image acquisition based on network synchronization. In: Proc. the 2006 conference on computer vision and pattern recognition workshop, pp. 1–6 (2006)
20. Mills, D.: Precision synchronization of computer network clocks. ACM SIGCOMM Comput. Commun. Rev **24**(2), 28–43 (1994). (http://citeseer.nj.nec.com/mills94precision.html)
21. IEEE standard for a precision clock synchronization protocol for networked measurement and control systems. IEEE Std 1588-2002, vol. 1, pp. 1–154 (2002-10-31)
22. Na, C., Obradovic, D., Scheiterer, R., et al.: Enhancement of the precision time protocol in automation networks with a line topology. In: Proceedings IFAC World Congr, Seoul, Korea (2008)
23. Obradovic, D., Scheiterer, R., Na, C., et al.: Clock synchronization in industrial automation networks: comparison of different synchronization methods. In: Proc. 5th Int. Conf. Informat. Control, Autom, Robot., Funchal, Portugal (2008)
24. Scheiterer, R., Obradovic, D., Na, C., et al.: Synchronization performance of the precision time protocol: effect of clock frequency drift on the line delay computation. In: Proc. WFCS, Dresden, Germany, pp. 243–246 (2008)
25. Noda, A., Yamakawa, Y., Ishikawa, M.: High-speed object tracking across multiple networked cameras. In: Proceedings of the 2013 IEEE/SICE international symposium on system integration, pp. 913–918 (2013)
26. Noda, A., Hirano, M., Yamakawa, Y., et al.: A networked high-speed vision system for vehicle tracking. In: 2014 IEEE sensors applications symposium, SAS, pp. 343–348 (2014)
27. Noda, A., Yamakawa, Y., Ishikawa, M.: Frame synchronization for networked high-speed vision systems. In: SENSORS, pp. 269–272 (2014)
28. Karthik, A.K., Blum, R.S., et al.: Recent advances in clock synchronization for packet-switched networks. In: Foundations and Trends® Signal Processing, vol. 13, no. 4, pp. 360–443 (2020)
29. Okabe, R., Yabuki, J., Toyama, M.: Avoiding year 2038 problem on 32-bit linux by rewinding time on clock synchronization. In: 2020 25th IEEE international conference on emerging technologies and factory automation (ETFA), vol. 1, pp. 1019–1022. IEEE (2020)
30. Waldhauser, S., Jaeger, B., Helm, M.: Time synchronization in time-sensitive networking. Network **51**, 51–56 (2020)
31. Hanasz, S., Kuklewski, M., Kasprowicz, G., et al.: Concept of an enhanced accuracy onboard time synchronization via communication link. In: 2020 IEEE aerospace conference, pp. 1–6. IEEE (2020)
32. Buhr, S., Kreißig, M., Protze, F., et al.: Subnanosecond time synchronization using a 100base-TX ethernet transceiver and an optimized PI-clock servo. IEEE Trans. Instrum. Meas. **70**, 1–8 (2020)
33. Khan, M.A., Hayes, B.: PTP-based time synchronisation of smart meter data for state estimation in power distribution networks. IET Smart Grid **3**(5), 705–712 (2020)
34. Chen, J., Yu, M., Dou, L.-H., et al.: A fast averaging synchronization algorithm for clock oscillators in nonlinear dynamical network with arbitrary time-delays. Acta Autom. Sin. **36**(6), 873–880 (2010)
35. Windows subsystem for linux 2.0 released microsoft. Retrieved June 30, 2020. [Online]. https://docs.microsoft.com/en-us/windows/wsl/
36. Wireshark 3.2.5 released. the wireshark foundation. Retrieved July 1, 2020. [Online]. https://www.wireshark.org/

Vasanth Subramanyam received his Bachelor of Engineering (B.E.) degree in Instrumentation and Control Engineering from Anna University, Chennai, Tamil Nadu, India in 2005 and his Master of Engineering degree in Electrical and Computer Engineering from National University of Singapore in 2008. He is currently pursuing his PhD in the Department of Electronics and Communication Engineering, National Institute of Technology, Jamshedpur, India. Since 2009, he is employed with Automation Division of Tata Steel, India and is currently Principal Technologist specializing in areas such as Machine Vision, Image and Video Processing, Internet of Things and Smart technologies.

Jayendra Kumar received his Master of Technology degree specializing in Electronics and Communication Engineering from National Institute of Technology (NIT), Jamshedpur and his Doctor of Philosophy from Indian Institute of Technology (IIT), Roorkee, India in 2019. He is currently an Assistant Professor in the Department of Electronics and Communication Engineering, National Institute of Technology, Jamshedpur, India and has numerous national and international publications specializing in the areas of Digital Image Processing, Machine learning and Embedded Systems. He has also been the Co-Chief Investigator of VLSI SMDP-C2SD project in NIT, Jamshedpur which was sponsored by the Government of India.

Shiva Nand Singh received his Bachelor of Technology degree specializing in electronics and communication engineering from Birla Institute of Technology (BIT), Mesra, India, in the year 1980 after which, he received his Master of science degree specializing in electrical engineering from Ranchi University, India in the year 1991. Thereafter, in 2009, he received his Doctor of Philosophy specializing in electrical engineering from the National Institute of Technology (NIT), Jamshedpur, India. Currently, he heads the Department of Electronics and Communication Engineering and spearheads research in the areas of VLSI, Microwave engineering, Solar cell technology to name a few. He has also been the Chief Investigator of VLSI SMDP-C2SD project in NIT, Jamshedpur which was sponsored by the Government of India.