International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-3, Issue-6, June 2017
ISSN: 2395-3470
www.ijseas.com

An Interoperable Synchronization Algorithm in Wireless Xbee Multimedia Sensor Networks

Sekata Olika Abdenaa [1], Dr. Ravindra Babu Bellam[1], Dr. Vasu pinnti[2]
Dean, College of Engineering, Wolaita Sodo University, Sodo, Ethiopia[1]
Associate Professor, Computer science dept. , Adama University, Adama, Ethiopia[1]
Associate Professor, WSU, Madhura nagar, vijayawada, Andra Pradesh, India[2]

Abstract –The synchronization problem of the wireless transmission in real-time visual surveillance applications was addressed. The distributed architecture of wireless multimedia sensor networks was adopted to obtain the key frames from multi-views synchronously, which has low computational cost and memory requirements. A good strategy for distributed computing was designed to assign tasks to specific smart devices. The main contribution of this architecture was a combination of wireless sensor networks and visual sensors. Image coordinates of detected objects are encapsulated in wireless data transmission. To track 3D localization in compute vision, the collaborative synchronization mechanism was presented to realize the synchronization of the key frames and asynchronization of the packet transmission from multi-views. The received data were divided into groups by the real-time clustering algorithm for 3D localization.

I. INTRODUCTION

For the disaster relief system, a reliable architecture of wireless multimedia sensor networks (WMSN) is important to various applications, such as people tracking, robot navigation etc. Because of parallel distributed sensor networks with wireless communication and redundant deployment, WMSN is one of the most promising and potential applications of self-organization, rapid deployment and information accuracy [1]. Owing to the limitation of wireless communication, the transmission of compressed images is troublesome and hard to receive real-time videos of multi-views synchronously [13]. In addition, privacy is a factor in a monitoring system, with no photographing. In this paper, smart visual sensors CMUcam5 (PIXY) are used to detect objects independently [17]. 3D localization in computer vision demands detected results from multi-views synchronously [2, 3, 8–10, 14–16, 18, 19]. Wireless sensor networks (WSN) are good at

cluster-based communication and ad-hoc network [7, 12]. Therefore, the combination of visual sensors PIXY and WSN is reliable to realize 3D localization. Owing to the high rate of false alarms in foreground detection [11], the proposed architecture employs color algorithm to detect objects [4].

The 3D localization in computer vision demands the time stamps of key frames from multi-views are synchronized. To the best of our knowledge, the synchronization of key frames and asynchronization of the packet transmission from multi-views are not well studied in WMSN. First, the coarse synchronization at ms level makes sure the local time of all sensors is simultaneous [3]. Second, to minimize the size of the packet, it encapsulates the image coordinates of the detected objects in the key frames [4] instead of the compressed images [13]. Third, the coordinator can receive the packets transmitted from multi-views at different time stamps. Fourth, to realize the synchronization of key frames obtained from multi-views, it employs a collaborative synchronization mechanism to sample the key frames at the visual sensors synchronously, and cluster the real-time received data obtained from multi-views in PC.

The main contributions of this Letter can be summarized as follows: First, for the sake of real-time data aggregation from multi-views synchronously, a combination of distributed XBee sensors network and visual sensors PIXY in CMUcam5 project has been proposed. Second, the collaborative synchronization mechanism was employed to sample and cluster the detected results synchronously in real time. Some experiments have been carried out in the real world scenario. Experimental results showed that the proposed architecture realized real-time data aggregation from multi-views synchronously and reliably.

II. 3D LOCALIZATION

XBee sensors network is used to establish the mesh network in Fig. 1. Eight visual sensors in two rooms comprise PIXY, Arduino UNO and XBee S2 sensors to establish the distributed architecture of WSN. The coordinator is connected directly to PC to exchange data through the serial port. All visual sensors are directed to the center of the rooms. The overlapping area of visual sensors can be extended accordingly. About 1.5 _ 1.5 m2 area of each room are overlapped by 4 visual sensors. The rest rooms are overlapped by less than 4 visual sensors.

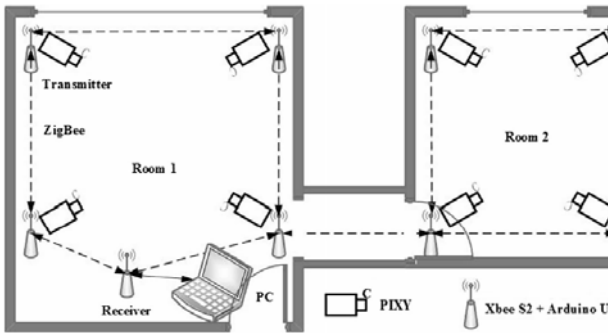

Fig. 1 The mesh network of the proposed WMSN

2.1 Distributed Computing

Real-time localization of detected objects is a daunting task, since it requires 2D image coordinates from multi-views synchronously. The time intervals among the 2D coordinates obtained from multiple visual sensors need to be limited to a few milliseconds.

It is not possible in [13], because it transmits compressed images in wireless communication. One sensor needs to exclude others in order to complete video transmission in a period of time. Distributed computing is put forward and depicted in Fig. 2. PIXY takes the responsibility

of object detection [4]. The size of API package, which includes 2D image coordinates of detected objects, can be reduced to only a few bytes. Therefore, Arduino UNO can complete the mission of encapsulating 2D location of detected objects into Application Programming Interface (API) package instantaneously. XBee S2 is responsible for the analysis and transmission of API packages. PC software is responsible for unpacking API packages, implementing real-time 3D common perpendicular centroid (CPC) localization

[6] And calibrating cameras.

2D image coordinate of the detected target in each view of monitoring cameras is the most important evidence to localize the target. The experimental platform of this Letter is shown in Fig. 1, the straight line between the 2D image coordinate and the target is the specific ray of one view through the optical center of the camera [6]. Between two rays of two views, the common perpendicular is the shortest distance.

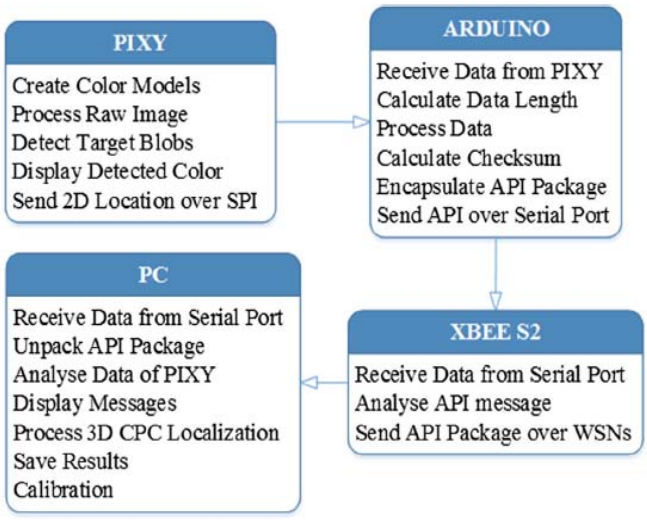

Fig. 2 Distributed computing among smart devices and PC

Collaborative Synchronization Mechanism

Furthermore, center point of the common perpendicular is the closest point between two rays. According to the theory of permutation and combination, four rays can be combined into 6 pairs of the rays. Six pairs of the rays can obtain 6 center points. The centroid of 6 center points is the closest point among 4 rays. Therefore, the centroid is closest to the theoretical intersection point of the rays. It is the estimated 3D location of the target, closest to the ground truth.

2.2 API Package Encapsulation

Depending on the configuration, XBee S2 can work in application transparent (AT) or API operating mode [5]. In AT mode, data is sent out through the serial interface when radio frequency (RF) data is received by the module. API mode is an alternative to AT mode. Data is transmitted in API frames. API specifies how commands, commands responses, and module status messages are sent and received from the module using serial interface.

The drawback of AT mode is that it is vulnerable to security attacks by sniffers like multi-channel packet analyzer. Figure 7 depicts captured packets by multi-channel packet analyzer. It clearly indicates the destination address, source address and mac payload, etc. However, API mode is a security mode for secure WSN routing protocols to ensure that the connectivity is maintained from security attacks. To implement RF transmission in API frames, the algorithms for API mode are indispensable for smart devices and PC. A radio firmware of XBee Zigbee must be installed in XBee modules. And, the software of XCTU is utilized to configure XBee modules to work in API mode and establish the XBee sensors network.

An algorithm of API package encapsulation is developed for the smart device of Arduino UNO in Fig. 2. The main process of encapsulation

I. PIXY sends the block information of detected results to Arduino at 1 Mbits/s over SPI. It should be 50 frames /sec. The most recent and largest objects are sent first. SPI is set to slave mode and rely on polling to receive updates. Due to RF transmission delay, all detected results cannot be transmitted to coordinator instantaneously. A threshold is employed to balance the speed of PIXY and the limitation of RF transmission. In order to meet the requirement of the algorithm in PC, the encapsulated data need to be formatted in advance. Checksum is also calculated to ensure data integrity and accuracy.

3. INTEROPERABLE SYNCHRONIZATION

The interoperable synchronization algorithm is performed on two levels:

(1) Coarse synchronization to synchronize sensors in the network and

(2) Collaborative synchronization to samples and clusters the detected results synchronously in real time.

At the coarse synchronization level, the coordinator sends the synchronization unicast packet to predefine visual sensors with a fixed period of 5 seconds. Compared with the broadcast packet in [3], the packet delivery acknowledgement can guarantee the communication quality. Because of the instability of wireless communication in the flood mechanism, visual sensors have to restart automatically sometimes. Periodic synchronization by the coordinator can make sure time stamps of all visual sensors are synchronized. At the collaborative synchronization level, visual sensors are responsible for the synchronization of sampling time for object detection and asynchronization of packet transmission

by wireless communication. PC is responsible for real-time clustering the received data into groups.

Since PIXY and Arduino are independent devices, it is a random local time when Arduino receives detected results from PIXY. If Arduino wants to detect messages delivery

from PIXY in every time interval a, it requires a tolerance time 2r. Furthermore, in order to keep only one message delivery from PIXY for each sampling, the time interval between h and h _ 1, successful detection messages delivery from PIXY, must be greater than 2r.

The time stamp of sampling by Arduino for detected results from PIXY is described as follows:

$$T_{sa}(\theta) = T_{sy} \quad if\ T_{sy} - T_{sa}(\theta - 1) > 2\sigma \cap (T_{sa}\%\alpha < \sigma \cup \mid \alpha - T_{sy}\%\alpha \mid < \sigma)$$

Where T_{sy} is synchronized time at the coarse synchronization level; h is a successful detection of message delivery from PIXY; σ is the tolerance time; α is the time interval for sampling by Arduino, the threshold a is a trade-off between the size of the transmit packet and capability of wireless communication; | . | is an absolute value sign.

Because of the efforts above, all visual sensors are synchronized. In the flood mechanism, visual sensors transmit packets to the coordinator synchronously. It causes the packet loss at the coordinator. Therefore, each visual sensor is allowed to transmit a packet in every 1 s. It means one packet content contains detected results of 1000/α key frames in the time interval α. Therefore, key frames of each

view in every 1 s form a video stream, which can guarantee the real time in a short time interval α. A unique time slot is assigned to each visual sensor in a predefined order. Each visual sensor is allowed to transmit a packet at its own time slot. The time stamp of transmission for first visual sensor is defined as follows:

$$T_t(l) = T_{sa}(\theta) + r(\omega) + l \times \beta \quad if\ T_{sa}(\theta)\%(2\times\alpha) \cup |\ (2\times\alpha) - T_{sa}(\theta)\%(2\times\alpha)\ |$$

Where r(ω) is a random value between 0 and 30, the upper limit of r(ω) is 30; β= 50 is the span for each time slot; l from 1 to 4 is the identity of visual sensors to label each sensor in a room. Therefore, the gap between two time slot is β-ω=20. In this order, the time stamp of received packets at the coordinator can be sorted in an array for all visual sensors;

% is modulo operation.

At the PC side, the received data need to be grouped by clustering algorithm in real time. In order to handle the fresh received data on time, the length of the segment for data clustering is defined as g=20 . When the fresh received data are cumulative to s=10 , a new segment is ready for clustering. The real-time data of a new segment is defined as follows:

$$D_{rt}(i) = D(i+m) \quad if\ i \in [1,g],\ e(t) - e(t-1) = m = e(t) - g$$

Where D is the received data by the coordinator; m + 1 is the first index of D in the tth segment; e(t) is the last index of D in the tth segment. When a new segment is ready for clustering, BFSN clustering algorithm is employed to cluster the real-time data into different groups, based on the time stamp of sampling by Arduino [8]. The clustering result is achieved by the following formula:

$$A = f_{BFSN}(f_{SimiR}(T_s(D_{rt})\%(10\times\alpha)), \gamma, \lambda)$$

Where fBFSN is BFSN clustering algorithm; fSimiR is used to create a normalized similarity matrix; Ts is the time stamp of sampling in the real-time data. The synchronization period of the coordinator is 5 s, which is 10 x α. As long as the time stamp of samplings in realtime data is a multiple of 10 x α, it means they are synchronized accordingly, in case that visual sensors fail to receive synchronization messages from the coordinator. The parameter which will derive the normalized similarity matrix is Ts(Dn)%(10xα) . It means that when the newly received data are cumulative to s = 10, a new segment is ready for clustering, the length of the segment for data clustering is g ¼=20 in (3). The clustering data is Ts, which is the time stamp of sampling. In order to filter out the unsynchronized data, Ts needs to use the modulus operator to find out the synchronized

data in the synchronization period of the coordinator, 5 s.

ϒ= 0.98 is a neighborhood threshold, if $f_{SimiR}(a,b)>ϒ$, a and b are neighbors. ϒ=0.98 is the classification threshold, if the number of elements in class B is num. There is a new element c need to be classified. If more than ϒx100 percent of elements num in class B are the neighbors of c, c is classified to class B.

However, the clustering result by BFSN algorithm is not following the rules of spatial and temporal correlation. In the practical scenarios, the rule of spatial correlation is that data received from different rooms need to be classified into different groups. The rule of temporal correlation is the data received at different periods of time that need to be classified into different groups, and the data received from the same visual sensor cannot appear at the same group. The real-time clustering algorithm based on spatial and temporal correlation is illustrated in Algorithm I:

Algorithm I: Real-time clustering algorithm

Input: D **Output**: tt_τ

1. $ttrt(1+m) = id$; $id = 1$
2. **for** i = 2 to g **do** %% Iterate through the array
3. **for** j = 1 to $sz(A)$ **do** %% Iterate through A
4. **if** $f(A(j) == i)$ **then**
5. $tti_A(i) = j$
6. **end if**
7. **end for**
8. **if** $tti_A(i)! = tti_A(i-1)$ **then**
9. $id = id + 1$
10. **end if**
11. $ttrt(i+m) = id$
12. **end for**
13. **for** i = 1 to g **do** %% Iterate through the array
14. **if** $vs(i) \in Ri_\tau$ **then**
15. $tt_\tau(ct_\tau) = [ttrt(i+m) \quad D(i+m)$
16. $ct_\tau = ct_\tau + 1$
17. **end if**
18. **end for**

where $Gi_A(i)$ is the class identity of ith real-time data defined by BFSN clustering algorithm A; f() returns the indices that satisfy the conditional expression; sz() returns the size of a matrix; Gn(i+m)= id is the group identity of the ith real-time data in the data of D; vs is the identity of the visual sensor in real-time data of Drt; Ri_T is a predefined vector of vs for the sth room; G_T is the final clustered data with the group identity and the sth room for the τth real-time data; ctτ is the last index of Gτ, ctτ is increasing according to the real-time clustering algorithm and ctτ=1 is the initial value. According to the rule of temporal correlation, if the class identity of the ith and (i-1)th real-time data are different, or the identities of visual sensors for the idth group do not contain the identity of visual sensor for the ith real-time data, the group identity id increase

1. Otherwise, id remains still. Based on the rule of spatial correlation, if the identity of visual

sensor for the ith real-time data belongs to Riт of the тth room, the group should be divided accordingly. Finally, each clustered group contains one or multiple unique views of a specific room.

The mean value of time differences is the key indicator for the performance evaluation of synchronization. To track 3D localization in computer vision, the mean value of time differences among the time stamp of sampling in each group should be as small as possible, which is defined as follows:

$$t^{\mu} = \frac{\sum_{i=1}^{n>1}\sum_{j=i+1}^{n>1}\sum_{k=1}^{\delta}\sqrt{(T_s(G_\tau(G_i(i)),k) - T_s(G_\tau(G_i(j)),k}}{\delta \times C_n^2}$$

$$if\ G_i = f(G_\tau == id)$$

Where n is the number of received data in a group, if n = 1, the mean value is 0; δ is the number of key frames in a received data, which is determined by the parameter α; Gi is the indices of received data that belongs to the idth group.

5 CONCLUSIONS

According to the requirement of the disaster relief system, architecture in indoor WMSN was presented. It established XBee sensors network and fulfilled the requirement of real-time data aggregation and RF transmission instantaneously. The algorithm results multi-views can be transmitted to the coordinator synchronously and clustered in real time. And also showed that the proposed architecture can realize real-time data aggregation of multi-views reliably and synchronously.

REFERENCES

[1] Akyildiz, I. F., Melodia, T., & Chowdhury, K. R. (2007). A survey on wireless multimedia sensor networks. Computer Networks, 51(4), 921–960.

[2] Berclaz, J., Fleuret, F., Turetken, E., & Fua, P. (2011). Multiple object tracking using k-shortest paths optimization. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(9), 1806–1819.

[3] Berger, A., Pichler, M., Klinglmayr, J., Po¨tsch, A., & Springer, A. (2015). Low-complex synchronization algorithms for embedded wireless sensor networks. IEEE Transactions on Instrumentation and Measurement, 64(4), 1032–1042.

[4] Dinh, H., & Inanc, T. (2009). Low cost mobile robotics experiment with camera and sonar sensors. In: American control conference, 2009. ACC'09 (pp. 3793–3798), IEEE.

[5] Faludi, R. (2010). Building wireless sensor networks: with ZigBee, XBee, arduino, and processing. Sebastopol: O'Reilly Media, Inc.

[6] Feng, S., Wu, C., & Zhang, Y. (2016). Passive target localization in multi-view system. In Control conference (CCC), 2016 35th Chinese (pp. 8444–8447). IEEE.

[7] Feng, S., Wu, Cd, Zhang, Yz, & Jia, Zx. (2014). Grid-based improved maximum likelihood estimation for dynamic localization of mobile robots. International Journal of Distributed Sensor Networks, 10(3), 1–16.

[8] Jiang-bo, Q., & Yi-sheng, D. (2004). A clustering algorithm based on broad first searching neighbors.

Journal of Southeast University: Natural Science Edition, 34(1), 109–112.

[9] Karakaya, M., & Qi, H. (2014). Collaborative localization in visual sensor networks. ACM Transactions on Sensor Networks (TOSN), 10(2), 18.

[10] Li, A., Lin, M., Wu, Y., Yang, M. H., & Yan, S. (2016). Nus-pro: A new visual tracking challenge. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(2), 335–349.

[11] Lim, C. H., Vats, E., & Chan, C. S. (2015). Fuzzy human motion analysis: A review. Pattern Recognition, 48(5), 1773–1796.

[12] Mantri, D. S., Prasad, N. R., & Prasad, R. (2016). Mobility and heterogeneity aware cluster-based data aggregation for wireless sensor network. Wireless Personal Communications, 86(2), 975–993.

[13] Pham, D. M., & Aziz, S. M. (2013). Object extraction scheme and protocol for energy efficient image communication over wireless sensor networks. Computer Networks, 57(15), 2949–2960.

[14] Shitrit, H. B., Berclaz, J., Fleuret, F., & Fua, P. (2014). Multi-commodity network flow for tracking multiple people. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(8), 1614–1627.

[15] Smeulders, A. W., Chu, D. M., Cucchiara, R., Calderara, S., Dehghan, A., & Shah, M. (2014). Visual tracking: An experimental survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(7), 1442–1468.

[16] Toni, L., Maugey, T., & Frossard, P. (2015). Optimized packet scheduling in multiview video navigation systems. IEEE Transactions on Multimedia, 17(9), 1604–1616.

[17] Wibowo, F. W., & Purwacandra, P. P. (2015). Object tracking using initial data to count object image based-on wireless sensor network. Advanced Science Letters, 21(1), 112–116.

[18] Yoon, J. H., Yang, M. H., & Yoon, K. J. (2016). Interacting multiview tracker. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(5), 903–917.

[19] You, K., Yang, W., & Han, R. (2015). The video collaborative localization of a miners lamp based on wireless multimedia sensor networks for underground coal mines. Sensors, 15(10), 25103–25122.