EventTube: An Artificial Intelligent Edge Computing Based Event Aware System to Collaborate With Individual Devices in Logistics Systems

Yijun Mo, *Member, IEEE*, Zhenchuan Sun, and Chen Yu, *Member, IEEE*

***Abstract*—Artificial intelligence has been adopted to facilitate monitoring, operation, and decision in the logistics field. Logistics robots with environment perception capability have been used to improve warehousing efficiency in logistics systems. However, autonomous mobile robots face computationally intensive and real-time demanding tasks such as navigation, localization, and obstacle avoidance. In this article, we present EventTube, an edge computing based event-aware system that can efficiently discover events from the video data captured by RGB-Monoculars and collaborate with individual devices to make timely decisions. EventTube deploys a semantic context extraction pipeline on edge servers to aggregate video streams from mobile robots and feed a few keyframes, including the start and end of the specific events to the successive perception pods, accelerating logistics robots' response speed. The event-related model parameters are trained and updated online on a server. The video data collected at the warehouse site for our mobile robots show that EventTube significantly improves parcel delivery efficiency without affecting regular deliveries.**

***Index Terms*—3-D detection and localization, artificial intelligence (AI), computer vision in logistics, edge computing.**

I. Introduction

With the rapid growth of logistics demand, artificial intelligence (AI) is widely used in the logistics field. The sorting, storage, and transportation of cargos in logistics warehousing have been made intelligent, significantly improving the function and corresponding quality of logistics services. Autonomous mobile robots (AMRs) with environment perception capability and sorting robots for sorting parcels are gradually playing a vital role in the warehousing segment of the logistics system.

The mainstream logistics robots have problems such as paths that need to be planned, inflexible obstacle avoidance, and rigid identification of cargo location and volume [1]. Environmental perception and sorting speed have become bottlenecks for logistics robots to improve efficiency. In the classic automated guided vehicle (AGV) and AMR automatic transport system, the video data captured by the logistics robot is directly transmitted to a high-performance server for the perception calculation of the robot's surroundings, and the server does intelligent planning after combining data from multiple sources to send down to the robot [2]. As a critical issue of logistics environment perception, the accurate detection of objects by robots can enhance their autonomous capabilities. The logistics robots employ high-resolution industrial cameras to monitor and capture the racks, obstacles, motions, and environments in a warehouse. The volume of video frames transmitted to the servers occupies massive bandwidth up to 1 Gb/s, while the bandwidth provided by most logistics warehouse networks is no more than 100 Mb/s. As we implemented on NVIDIA 2080Ti, a high-resolution frame's average decode and per-event analysis delay cost about 50 ms. The unnecessary communication and computation introduce significant event detection delay that increases linearly as the increment of events in the frame. After dozen frames, the delay accumulates rapidly, causes the logistics robot to take serious lagging actions, and dramatically reduces the transportation efficiency. Of course, it is also impossible for AMRs to perform the high-resolution object detection, localization, and analysis independently due to the lack of sufficient computation capacity.

Frames skipping and sampling are adopted to reduce bandwidth and computing overhead intuitively, but that is very limited and loses event frames. The logistics robots have apparent target objects and events in specific environments and locations, which are sparse in the video stream. Extracting the regions of interest (ROIs), sizes, locations, tracks, relationships, appearance times, and other semantics attributes of the small part of objects will significantly reduce the volume of spatio-temporal data. At the same time, the logistics robots can only complete simple tasks of the whole semantics extracting pipelines; other computation-intensive tasks should resort to the edge servers around AMRs.

Manuscript received 16 February 2022; revised 10 June 2022; accepted 30 June 2022. Date of publication 7 July 2022; date of current version 13 December 2022. This work was supported by the National Key Research and Development Program of China under Grant 2020YFB1805203. Paper no. TII-22-0771. *(Corresponding author: Chen Yu.)*

The authors are with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: moyj@hust.edu.cn; szc_wtx@hust.edu.cn; yuchen@hust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2022.3189177.

Digital Object Identifier 10.1109/TII.2022.3189177

The accuracy and computation of semantics extracting affect the performances of the following autonomous capabilities; the edge servers must trade off between the accuracy and computation complexity via overcoming three challenges. First, the edge servers should switch among ROI detection models according to environment context to improve the accuracy with low computing complexity while predicting events type and start and end times. Second, the neural network models chosen by the edge servers must be lightweight enough to handle more streams from AMRs with lower resources and delay. Third, the edge servers should face various scenarios with or without events and the events captured by static or moving cameras.

By event-triggered semantics extracting and data acquisition methods, the communication overhead can be reduced, the speed of target monitoring and tracking can be increased, and the responsiveness of the whole autonomous driving system can be improved. The key to improving the transportation efficiency of logistics robots is to improve the ability to obtain information from the environment. Existing systems have used 1) LiDAR-based vehicle localization and environment sensing methods [3], which are accurate and efficient but costly and 2) event-aware RGB video analysis methods, which locate events through the changing relationship between video frames [4] and then analyze the set of frames where events occur.

Although edge servers take on most of the task, AMRs also need to fully use their capability of video compression, shallow neural network, and weak accelerating to reduce network bandwidth overhead coarsely. Otherwise, the backhaul video streams produced by the amount of AMRs in the local area will block the network. In the article, we argue that the difficulty in achieving environmental state awareness on AMRs for logistics needs is not a lack of neural network capability. Our central insight is that the inability to simplify neural networks to run on AMRs and their RGB sensors is not an inherent problem of neural networks but rather an underutilization of the properties of RGB video streaming in logistics transportation. For example, the standard mobile transport device upstream data-aware approach FilterForward trains the model directly on the entire video stream, and even with the strategy of simplifying the model, the model still requires resources well above the end device hardware limit. Furthermore, obstacles captured by AMR's cameras usually appear as moving regions in consecutive frames, hit detection relies on interframe variation, and high-precision analysis relies on analyzing a segment of the video, including temporal information. We assert that 1) a fast event detection model based on inter-frame variation and timing information can best represent the distribution characteristics of events in RGB-Monocular video, 2) Low-level feature screening methods can achieve a high event hit ratio and high accuracy in continuous frame sequence analysis. Therefore, these simple neural networks can be surprisingly effective at environment perception (more effective than techniques for compressing models trained on large datasets!) Based on the two assertions, our proposed event consists of three parts: the diff module is installed on AMRs to focus on moving objects, and their lower level features whether the robots are static or not and optimize the backhaul transmission overhead; the tubelet module is performed on edge servers to extract semantic event tubes from moving objects related video streams and feed the result to following pipelines on demand; the online training and posterior module are run on a central server to optimize the model parameters of the diff module and the tubelet module.

The contributions of this article are as follows.

1) We have proposed and implemented EventTube, an environment-aware system for logistics and transportation. The core tubelet module is executed on edge servers to filter out semantic event-tubes for improving the accuracy of AMRs' environment perception and speeding up AMRs' operation efficiency.
2) The EventTube adopts shadow and sparse neural networks to locate ROIs of event classes from differential temporal sequence obtained by diff module, which is lightweight with small computation. We also introduce a back propagation mechanism to adjust the start time, end time, location of ROI tubes, and following pipelines on demand. The mechanism provides reliable detection results even for videos taken on low-cost monocular cameras.
3) We also employed an online training method to adapt to various scenarios, whether AMRs are static or moving. Experiments on the logistic activity recognition dataset (LARa), KITTI dataset, and real scenario showed that the EventTube could improve event hit rate and reduce bandwidth overhead and overall delay. The EventTube improves transportation efficiency without affecting the standard delivery of packages.

The rest of this article is organized as follows. We discuss related work in Section II. Section III provides the system model and problem definition. In Section IV, we present EventTube, an edge computing based RGB camera-aware system that can efficiently acquire data from video data. Then, we present the performance evaluation in Section V. Finally, Section VI concludes this article.

II. Related Works

A. Environment Perception Methods for Mobile Vehicles

1) LiDAR-Based Method: Considering the detection and localization accuracy of object, AMR is currently mainly guided by LiDAR-based methods. Researchers have used high-precision LiDAR point clouds for accurate 3-D object detection and corresponding localization. PointNet [5] demonstrates the effectiveness of the direct application of CNN on point cloud through experiments and the stability of the network through theory. The work [6] achieves high accuracy results by converting the point cloud of LiDAR data into pixel-level depth information and combining it with RGB video data using the commonly used CNN method. The work [7] established an observation model of AGV based on LiDAR to quickly obtain the accurate position of the vehicle by matching the observation information and achieving the environment awareness during autonomous localization. The work [8] implemented an AMR robot on TurtleBot3 Burger

by combining 2-D LiDAR, RGB-D camera, and related safety devices. LiDAR has some drawbacks, such as high cost and sensitivity to adverse weather conditions. These limitations suggest that employing LiDAR-based object detection and 3-D localization system is unrealistic in practical, day-to-day applications. Conversely, RGB cameras are relatively cheap, ubiquitous, and can potentially be resilient to most environments.

2) Video-Based Method: Cameras provide detailed information in the form of pixel intensities, which can reveal shape and texture properties on a larger scale. The work [9] fused RGB-D cameras with wheel odometry to construct a localization system with a field-tested localization accuracy of centimeter-level on the dataset without precalibration. Recent works have explored the prospects of 2-D RGB images for 3-D detection. The work [10] presents a method for 3-D object detection and pose estimation from a single image. The work [11] uses a multiframe optimization technique to achieve depth information extraction utilizing the camera's self-motion on the time series, which improves the accuracy and consistency of 3-D localization results achieving results comparable to LiDAR even on RGB-monocular. The work [2] transmits the video data collected by the logistics robot to a high-performance server for environment perception calculation. The server does intelligent planning after synthesizing data from multiple sources and sends it to the robot. The cost of monocular cameras is low, but due to the computational overhead of detection and localization tasks on video, it cannot be promoted on a large scale.

B. Edge Computing and Collaboration in Logistics Systems

1) Video Event Acquisition on Mobile Robots: This class of approaches considers that frames are dense and events of interest are sparse in the video stream captured by the camera. The idea of feature-based frame filtering can be widely seen in the CV community, and many of these approaches are used to retrospectively classify or identify events in video [12]. Fast filtering system for video analytic (FFS-VA) [13] achieves a reasonable tradeoff between throughput and AdaFrame [12] training of long- and short-term memory networks to adaptively select frames with important information. Tour into the video (TIV) [14] considers deep neural network (DNN) split inference in industrial scenarios, where edges run as many layers as possible before sending intermediate values to the cloud. Reducto [4] implements a practical frame filtering system on cameras at low computing power by modeling the relationship between feature type, threshold, and query accuracy. In contrast to all these solutions, EventTube targets high recall high filtering of real-time filtering on end devices whose resources can support micro-NN networks. DNN-driven streaming (DDS) [15] drives data acquisition through the receiver, gets feedback on low-quality data, and transmits high-quality data for a portion of the region based on the input. In this article, we investigate that ROIs are particularly fixed in logistics processes; so EventTube integrates a video ROI method based on server feedback based on frame filtering to reduce computational and transmission further overhead.

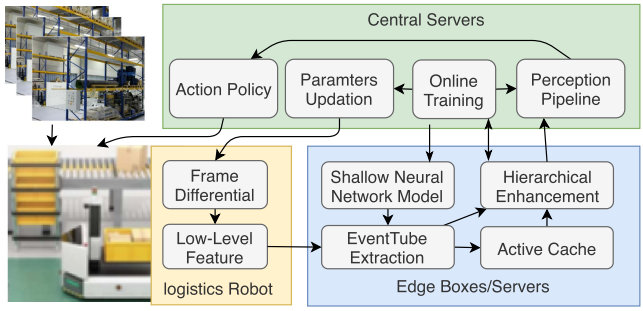


Fig. 1. System overview.

2) Control Methods for Logistics Robots: In logistics systems, a collaboration between devices can improve robot positioning accuracy and transportation efficiency. Chen *et al.* [16] implement AGV positioning and cargo status recognition on a global surveillance camera and controls logistics robots for the cargo loading, unloading, and transportation by using data network communication technology. Zhang *et al.* [17] investigate the collaboration of multiple mobile robots in a factory freight logistics system, construct a panoramic view by the air–ground cooperation to overcome the enormous difference between the aerial view and ground view, and achieve high precision collaborative robot positioning. Those collaborative control methods were dependent on a global panoramic view generated from most video streams in a warehouse, which achieved high precision at the expensive cost of stream synchronization, network bandwidth, and view synthesis. In this article, the EventTube distributes the basic preprocess tasks of video streams to edge servers and feedback ROIs and their timestamp to a central server for collaboration, which only requires simple time alignment and very little computation overhead for coordinate transformation and object fusion. The previous collaboration systems can be strengthened and put into fact in a low cost.

III. System Overview and Definitions

Fig. 1 depicts the architecture and main modules of our proposed EventTube system. Currently, EventTube supports mobile robot obstacle detection, cargo identification, and localization tasks in the logistics field. These video information acquisition tasks are described in Section V-A.

A. Pipelines on Central Servers

In a new scenario, initial models and parameters of AMRs and edge servers do not work well for accurate logistics perception, which should be pretrained on the powerful central server. In the beginning, the original video stream generated from an AMR is directly forwarded to the central servers by EventTube. The server runs a traditional video analysis pipeline in a few minutes of video to get the characterization of the video. The prediction result of an event tube is directly related to whether the frame is a

keyframe or not. Therefore, we make the traditional video analysis pipeline process each video frame and get the corresponding prediction result. We collect all the feature data and results in this video, find the most suitable interframe differential features with corresponding thresholds, design each frame trigger interval value for the tubelet as a regression label, and train the tubelet. At the end of this phase, the most suitable low-level features for each perception task are identified and stored on the server, the suitable tubelet regression model for this video stream is deployed in the terminal, and the initialization of the weights is completed.

B. Diff Module on AMRs

The Diff module extracts interframe differences and low features. Due to the scaling or moving of AMRs' camera, the module calibrates successive frames to the exact coordinates by spatial sampling, padding, and cutting before frame pixel differential. After frames alignment, the module will perform three tasks, per-frame feature extraction, interframe feature difference calculation, and filter. After receiving a query request, the server analyzes and derives the best features and filtering threshold for the query task and informs the interframe differencing module. After receiving the configuration information from the server pipeline, the feature extractor calculates the specified frames' specified features and continuously extracts the differences in the characteristics between tracking consecutive frames. The computed difference value is compared with the filtering threshold for each pair of consecutive frames. Frames with a difference value less than the threshold mean that only the same result is obtained even if both frames are uploaded to the traditional video analysis pipeline, which indicates that filtering out one of the frames does not affect the accuracy.

C. Tubelet Module on Edge Servers

The tubelet module is the core part of the EventTube system. It adopts hierarchical lightweight shallow convolutional neural network models to generate tubes and their semantics from interframe differences and low features, then only transmits the coarse event type and filtered ROIs' features to the final perception and action. The head model is a binary classification model to justify if any event exists. The middle model is a prediction model for the start and end of the event. Meanwhile, the prediction model is also verified on time by receiving features from the Diff module. Finally, the tail model is an event spatial positioning model used for ROIs' adjustment. To train the models and parameters, the central server processes all frames, calculating the interval value between each frame and the starting point of the frame of interest for the tubelet as a regression tag.

The central server uses a model trainer, which quickly trains a simple regression model for each input that characterizes the relationship between frame features, the current frame moment, and the first keyframe in the future. The simple hierarchical regression models are trained based on a shallow convolutional neural network, which typically takes only a few seconds to train on a few minutes of video. The models deployed on the edge servers run on a compressed set of consecutive frames, which can accurately predict the ROIs in a future sequence of frames combining with a timestamp and temporal features and provide higher filtering power. At the same time, the model is simple enough to meet the real-time requirements even on the end device.

D. Hierarchical Frame Enhancement Module

The state changes on the video stream are continuous and not abrupt, and the frames in the over the state can also provide information during analysis; so the video sequence needs to be provided to the server for analysis. The hierarchical frame enhancement module on the terminal reduces the amount of data that needs to be transmitted through ROI cropping and hierarchical enhancement methods. The ROI region is obtained based on the results of the server's reasoning in analyzing the pipeline over a few minutes of video. The hierarchical enhancement method sets different levels of compression for other parts of a sequence, choosing a high compression rate at the beginning of the state and gradually decreasing the compression rate during a series, ending with HD frames. The compressed frames come from the active cache.

E. Threshold Tuning and Model Retraining

After the terminal receives the specified low-level features and difference thresholds and the tubelet receives the regression model trained by the server, EventTube starts filtering the frames. However, in some cases, the video dynamics may change (e.g., lighting changes), and the existing difference module and tubelet still analyzed originally may lead to drift phenomena. In this case, the conditioner periodically sends the frames to the model trainer on the server and updates the model, and the server sends the updated differential thresholds and regression model to the endpoint. The mismatched values and the corresponding original frames are also sent to the model trainer in case of incorrect predictions. It is worth noting that the update period is not fixed, but the period size is adjusted based on the prediction confidence.

F. Active Cache

Sending mismatched values and raw frames means that the end device needs to provide additional storage to cache frame data over a period of time. Taking advantage of this, we design an active cache that improves efficiency and accuracy. As mentioned in Section III-E, the server has a dynamic feedback period, and as discussed in Section III-C, the tubelet needs to take into account timing information. The traditional analysis pipeline on the server also needs to utilize a video sequence to analyze effectively. Which frames do we use as information in a timing sequence? How many frames are selected? We propose an adaptive sampling strategy where the cache size is aligned with the total number of frames in the dynamic feedback period. The frame sampling strategy is described in detail in Section IV-C.

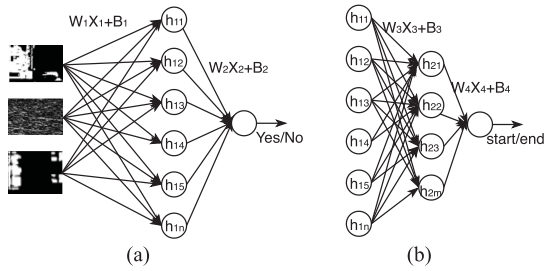


Fig. 2. Hierarchical shallow neural network. (a) Classification model for event existing or not. (b) Classification model for event start or end.

We verify the impact of the number of frames selected on EventTube by comparing the experiments.

IV. Our Proposed EventTube

A. Hierarchical Shallow Neural Network

EventTube simplifies the complex multiclassification semantic extracting task into a series of hierarchical binary classification tasks. The hierarchical shallow neural networks infer the existence, start–end points, and location of events in sequence. The models are executed in parallel as successive pipelines, and the outputs of previous models' hidden layers are fed into the following models as input. The hierarchical mechanism makes the followup models share the computation with former models. Fig. 2 shows that the start–end identification model infers based on the output of hidden layer of event-existence detection model, which increases the depth of successive model, but avoids unnecessary reinferring and duplicate copies. After complete inferring, the models obtain event tubelet with full semantics and feed them to the central server for final control policies. To improve the hit rate of hierarchical binary classification models, when quiet time with no events exceeds the threshold, the EventTube informs AMRs to transmit full-frame samples to the central server, entering the incremental training phase.

B. Event Localization With Tubelet

Low-level feature-based interframe differencing aims to identify critical frames that may affect the results by capturing frames with significant feature changes. But the interframe differencing filter itself does not understand the content of the frames, which makes the potential filtering power not strong. The tubelet module applies neural networks Section IV-A to mine the semantics of frames and consider the relationship between consecutive frames on a time series to achieve more powerful filtering with guaranteed hit-rate and recall of event tubelet.

Specifically, we design a simple NN-based regression model that runs on consecutive compressed frames, first extracting the image semantics by a simple convolutional neural network (or operator in the computer vision community), then combining the semantics of multiple frames, and finally feeding the pooled features into a linear regression network to predict the score. The frame difference module in Section III-B can detect the interframe diff-value, and when the diff-value is large, it can be considered that there is sufficient semantic information on the time series of that segment, which is likely to affect the analysis result. Based on this, we can capture all frames with different values more significant than a threshold over a period of time, which can significantly characterize the state migration process of the content in the video stream over the time series. We input the frames filtered by the frame difference module into the tubelet, which combines the timing information and predicts the location of the start/end moment of our event tube of interest through a regression model. This is used as a criterion to continue filtering out useless frames without reducing the recall rate. When the tubelet module detects the event tube, it will cache both the frames of the excessive state and the event tube as key frames $\mathbb{F}$ in the active cache and set the return parameters of the event tube according to the server configuration (see Section IV-C).

It is worth noting that the tubelet model is simple, only suitable for specific conditions. The video dynamics may change (e.g., lighting changes), and the existing differential module and tubelet are still analyzed according to the original method and may lead to a drift phenomenon. Therefore, we will update the model in the tubelet module according to the period T. The value of T is not fixed and will change as the confidence level of the tubelet module results changes. Regular online updates of neural networks are a challenge. In contrast to standard deep learning algorithms, training a model online means that the frames of the input model are not independent and identically distributed, especially on relevant streaming data. In this case, the general neural network quickly forgets the memory of the original data while adapting to the new frames.

As we observed in Section I, if a critical frame exists in the video data, the distribution density of positive samples close to the key frame is high for a period of time. Based on this, we propose a systematic online update strategy. Specifically, we set recall constraints R and define the maximum and minimum periods $t_{\max}$, $t_{\min}$. We initialize T to $t_{\min}$. The tubelet module performs inference on the frames filtered by the difference module. When the number of frames is a multiple of the current interval T, the system uploads it directly, even if tubelet prefers to discard it. The server evaluates the tubelet result p_{tube} and the actual predicted result p_{tl}. If the regression deviation is within the constraint, the interval T is increased by 1 s

$$T_{i+1} = \begin{cases} t_{\min}, & R_i < R; \\ \dfrac{T_i}{2}, & \text{conf}_i \le \text{conf}_{\text{std}}; \\ T_i, & \text{conf}_i > \text{conf}_{\text{std}}, T_i = t_{\max}; \\ T_i + 1\ s, & \text{conf}_i > \text{conf}_{\text{std}}, T_i < t_{\max} \end{cases}. \quad (1)$$

Otherwise, T is halved, and if the recall is lower than the query's constraint due to the wrong result of the tubelet, the period is reformulated to $t_{\min}$, and multiple iterations are executed. This dynamic cycle update strategy is commensurate with the dense distribution of positive samples in the time-dependent video stream. As our evaluation shows, we can train online learning tubelets with high recall and filtering rates, even using standard gradient descent optimizers.

C. EventTube Profile

The server runs a traditional video analysis pipeline. In the initial stage, the trainer uses a few minutes of representative video frames to compute each frame and obtains the detection results, including those using the traditional video analysis pipeline, the difference value of each candidate feature, and the pretrained tubelet model. In contrast to the traditional pipeline, we do not cause the server to perform inference on each HD frame at moments other than the initial stage, but only on the filtered event tube.

We deploy an active cache on the endpoint to load the data in the event tube. A small active cache would result in insufficient information in the frame sequence and would not bring richer semantics to the detection task compared to high list frames, while a too-large active cache would result in more load on the endpoint and high uplink overhead. We use a hierarchical enhancement algorithm, as shown in Algorithm 1. The component on the end device provides some initial parameters, including the set of filtered frames $\mathbb{F}$, the collection of incremental differences $\{r_i\}$ provided by the interframe difference component, and the period T supplied by the tubelet module component. The component provides other parameters, including the number of frames in the encoding sequence to be uploaded N, and the hierarchical enhancement configuration in the encoding sequence cfg. We regulate the active cache size in terms of two scales, the active cache sequence length m, and the number of frames in the uplink sequence N. Regarding the choice of m, we aim to preserve the state change process of a sequence as possible and, therefore, set m to match the total number of frames in the dynamic feedback cycle of the tubelet. By calculating the product of period T and frames per second (FPS), we can obtain the total number of frames m in the cache. After determining m, we should perform appropriately spaced sampling from m frames to generate further sequences. The server dynamically adjusts N and cfg according to the balance between resources and precision.

The goal of the selection is to preserve as much of the process of state change in the sequence as possible, and the interframe difference module has calculated the interframe difference d_i for the specified feature (d_i denotes the result of the difference between frame i and frame $i+1$ in sequence m). We select frame f_l such that the $\sum_{i=0}^{l-1} d_i$ and $\sum_{i=l}^{m-1} d_i$ of the interframe differences on both sides of f_l in the sequence is minimized, obtaining the subsequences m_1 and m_2, and so on until the first N frames are selected. The first N frames are arranged in chronological order, and the encoded sequence $\{f_i'\}$ is generated based on the compression parameters configured by the server. The $\{f_i'\}$ is placed in the uplink and uploaded to the segment analysis pipeline considering the timing channel information. The server in EventTube will see a pyramidal sequence of frames and analyze them based on that. When using the timing information to analyze the cargo or obstacle state in the video, not every frame in the sequence needs to be in HD, but only the change of state needs to be represented, and the compression rate of the frames can be adjusted to reduce the communication pressure. We believe that compressing some frames in the sequence does not affect the recognition accuracy of the server-side, and the smaller the index of the n frames uploaded, the higher the compression ratio can be set.

Algorithm 1: Hierarchical Enhancement Algorithm.

Input: $\mathbb{F}, \{r_i\}, T, N, cfg$
Output: $\{f_i'\}$
Init PriorityQueue Q_{seq}
$m \leftarrow T \cdot fps$
$\{f_m\} \leftarrow$ slice $\mathbb{F}$
PUSH($Q_{\text{seq}}, \{f_m\}$)
for $n \leftarrow 0$ to N **do**
  if IS_EMPTY(Q_{seq}) **then**
    BREAK
  end if
  $\mathbb{F}' \leftarrow$ POP(Q_{seq})
  $s, e \leftarrow$ GETBOUND($\mathbb{F}'$)
  mid $\leftarrow r_s + (r_e - r_s)/2$
  $l \leftarrow$ BINARYSEARCH($\{r_i\}$, mid)
  PUSH($Q_{\text{seq}}, \{f_s, \ldots, f_l\}$), PUSH($Q_{\text{seq}}, \{f_l, \ldots, f_e\}$)
  INSERT($\{f_i'\}, f_l$)
end for
for $f_i' \leftarrow$ POP($\{f_i'\}$) **do**
  $f' \leftarrow$ COMPRESS(f_i', cfg)
  INSERT($\{f_i'\}, f'$)
end for
return $\{f_i'\}$

V. Performance Evaluation

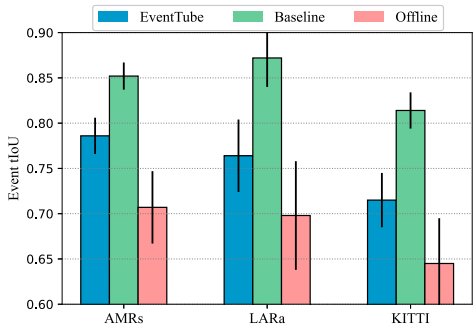
A. Experiment Settings

1) Data Sources and Analysis: Our dataset consists of video data captured on RGB-Monoculars from mobile robots in logistics warehouses, LARa [18] and public autonomous driving dataset KITTI [19]. The logistics warehouse site covers various obstacle and movement scheduling scenarios, and the video quality varies depending on the movement speed and lighting conditions when the RGB-Monoculars are capturing the data.

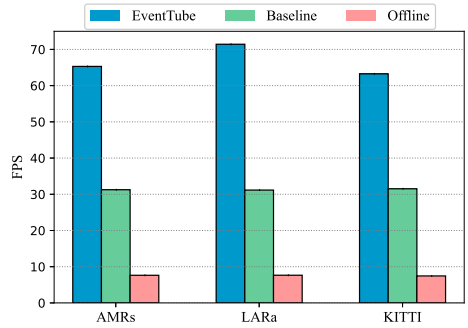
2) Query Tasks: Automation of warehouse goods transportation, e.g., AMR automatic obstacle avoidance, logistics robot positioning, etc. Automatic sorting of goods in warehouses, e.g., goods status recognition.

3) Experimental Environment and Server Model Setup: EventTube modules ran on AI chip based edge server with 0.6TLOPS NPU, 1.5 GHz CPU. The central server ran on a Centos instance with 32 CPU cores and 1 NVIDIA 2080Ti GPU. The camera was a virtual machine (VM) whose resources were provisioned based on [4]. In the VM scenario, we limited the RAM to 512 MB, the CPU to a single core, and the CPU frequency to 2 GHz via cpulimit.[1]

[1][Online]. Available: https://github.com/opsengine/cpulimit



(a)



(b)

Fig. 3. Comparing EventTube and the other two strategies for the datasets (a) event tIoU and (b) achieved FPS.

B. Overall Performance

We chose two baseline algorithms to verify the effectiveness of EventTube.

1) *Baseline:* We first compare it to a baseline video analysis pipeline where the AMR does not perform any operations and transmits the captured data to the server for analysis. This approach demonstrates the superiority of EventTube in terms of response time and bandwidth savings.
2) *Offline:* To verify the importance of context in video information acquisition, we compare EventTube with an optimal offline system that uses a detection model [20] based on running on a single frame to discover events. This approach demonstrates the superiority of EventTube in terms of event response speed and accuracy (i.e., event hit ratio).

We define time domain Intersection over Union (IoU) (tIoU) to evaluate the accuracy and efficient of semantics predict, which reflects the start–end gap between evaluation and groudtruth as (2), where f_s^g, f_e^g are start and end frames of groudtruth, f_s^p, f_e^p are start and end frames of prediction, and count(f_s, f_e) is count of frames between f_s, f_e

$$\text{tIoU} = \frac{\text{count}(f_s^g, f_e^g) \cap \text{count}(f_s^p, f_e^p)}{\text{count}(f_s^l, f_e^l) \cup \text{count}(f_s^p, f_e^p)}. \quad (2)$$

Fig. 3(a) shows that EventTube achieved an tIoU of 71.5%–78.6% in our selected dataset. The event hits indicate that the mobile robot can detect the corresponding cargo or obstacle with the data collected by RGB-Monocular. As expected, the tIoU of

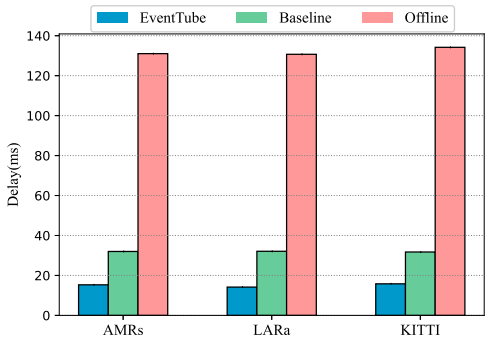


Fig. 4. Response delay comparison of EventTube and the other two strategies.

EventTube is significantly improved compared to the single-frame detection model.

Despite this high tIoU, EventTube can always provide a detection delay more negligible than the other two systems. As illustrated in Fig. 3(b), EventTube achieves a fast response to events by prescreening redundant data using a diff module and then predicting the start and end moments of events using a regression model while maintaining a tIoU.

1) Response Time: The promise of frame filtering is ultimately to reduce resource overheads and deliver (highly recall) query results with low latency. Fig. 4 illustrates that EventTube is able to reduce median per-frame response times by 44.15%–49.82% (14.17–15.81 ms) compared to the baseline pipeline.

On the other hand, due to the simplicity of EventTube and average bandwidth saving, it has not brought more reasoning delay and communication based on successfully reducing the number of frames the server needs to process. That makes the back-end processing speed increase by above 60%.

2) On Resource-Limited Devices: In our experiments, the resource-constrained VMs are considered as the camera component of the video analysis pipeline. Specifically, we chose a VM instance with a 2-GHz single-core CPU and 512-MB RAM. This profile follows the range of camera resources observed in the commodity cameras and surveillance deployments studied in [4]. We implemented EventTube on VM instances using Opencv and Pytorch for low-level feature extraction and continuous frame classification, as well as for threshold adjustment and training window scaling.

EventTube runs at 67.5 fps in a constrained environment, highlighting the ability to perform real-time filtering. By digging further, the processing overhead of different parts of EventTube is shown in Table I. Most of the processing overhead in the frame difference module is due to the per-frame extraction running with Python and OpenCV. Even if no processing is applied, the fps per frame extraction is 125.8. The low-level feature filtering approach also only achieves 107.4 fps compared to the tubelet module running at 78.3 fps. Overall, we observed that the filtering results of EventTube for each video exactly matched the results of our VM-based implementation, which illustrates the feasibility of EventTube for large-scale applications on AMR and sorting robots.

TABLE I
COMPARISON OF THE INFERENCE SPEED AND EVENT HIT RATIOS OF EACH COMPONENT AND ITS COMBINATION IN EVENTTUBE

EventTube components	Event hit ratio	Bandwidth (MB)	FPS
Low-level Feature	–	–	125.8
Frame Difference	–	–	136.7
Diff module	95.4%	2.43	107.4
Tubelet module	92.7%	0.913	78.3
EventTube	95.2%	0.872	67.5

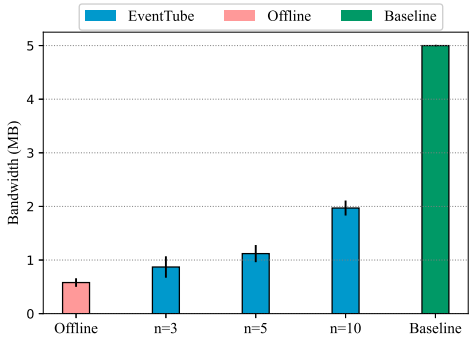


Fig. 5. Bandwidth overhead for our dataset. The n is the number of frames in enhancement sequence (see Section III-F).

3) Bandwidth Saving: Because EventTube considers timing information, it aims to monitor an event tube instead of a single frame. Therefore, EventTube can find possible events of interest more quickly by observing the migration status in a video context, effectively reducing the response time. The response time is mainly composed of communication delay. Fig. 5 shows the bandwidth savings brought by EventTube. 4 Offline YoloX infers the frames on the edge servers directly and then only sends the results to a central server. Offline processes only quarter frames in time with less bandwidth but more delay, which causes important events to be missing and requires more edge servers. Under the requirement of 90% recall by relying on higher filtering capabilities, EventTube still saves about 60% (even when $n = 10$) of the bandwidth. The main overhead is to transmit the compressed frame sequence.

4) Maximum Number of Connected Logistics Robots: According to our collected warehouse site dataset, the communication capacity in logistics warehousing is limited by the total uplink limitation (usually 10–100 Mb/s). Taking the uplink bandwidth limitation as an example, we tested the maximum number of camera links under three different systems, EventTube, baseline, and offline, by simulating the collected warehouse video dataset. As shown in Fig. 6, the maximum number of parallel cameras supported by EventTube under the 10-MB bandwidth limit is five times higher than the baseline. That means the same logistical efficiency can be achieved with one-fifth of the bandwidth cost without compromising the quality of work per device, compared to an end-to-end detection approach that does not use optimization. Alternatively, EventTube can place more AMRs and AGVs in logistics warehouse centers that already have the infrastructure to increase freight efficiency dramatically. As bandwidth increases, the gap between EventTube and

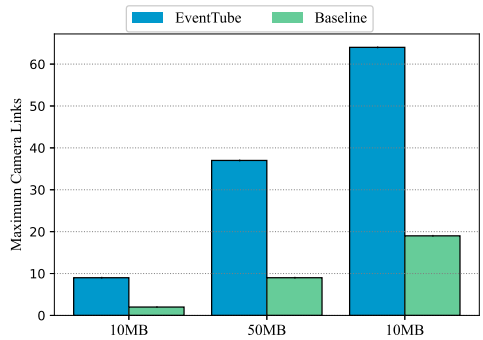


Fig. 6. Parallelized testing under bandwidth constraints.

the baseline and offline approaches shrinks because "Events" on multiple devices can be concentrated in the same time slot, which is the bottleneck for parallelization improvement at the most congested moments.

C. Comparison With Other Detection Methods

We also compared EventTube with three existing detection approaches that can consistently meet the desired accuracy target.

1) YOLOX-Tiny [20]: We considered an event detecting system that computes approximate query results using a compressed detection model. We trained a YOLOX-Tiny model on the AMR-monocular video dataset to detect events encountered by mobile robots in warehouses and then tested it on the remaining video clips.

2) Geometry [10]: Geometry is a method for 3-D object detection and poses estimation from a single image. It first regresses relatively stable 3-D object properties using a deep convolutional neural network and then combines these estimates with geometric constraints provided by a 2-D object bounding box to produce a complete 3-D bounding box.

3) LOCNet [11]: LOCNet is a monocular camera 3-D detection and localization framework that integrates 3-D based object detection, tracking, and localization. Through multiframe optimization techniques, mining depth information in video streams captured by monocular cameras outperforms state-of-the-art image-based detection schemes in various scenarios. In our experiments, the mean Average Precision (mAP) of Geometry and LOCNet have compared on the KITTI(easy) dataset. It is important to note that neither Geometry nor LOCNet can be deployed to AMRs, despite their efforts for real-time analysis.

Table II shows that EventTube achieves significantly greater benefits compared to the three systems. On storage video captured by AMR cameras, EventTube's event hit rate was three percentage points higher than the YOLOX-Tiny method running on the same underresourced device, while the FPS was twice as high. Because EventTube uses a diff module that detects events quickly and filters out a large amount of redundant data before performing event analysis, this means that EventTube can improve AMR productivity. We tested EventTube against current monocular camera 3-D target detection frameworks on the publicly available dataset KITTI(easy). Although the accuracy is not significantly improved compared to the currently

TABLE II
COMPARING EVENTTUBE WITH EXISTING DETECTION AND LOCALIZATION METHODS IN SMART LOGISTICS SYSTEM

System	AP	Bandwidth (MB)	FPS
EventTube (custom video)	95.21%	0.872	67.5
YOLOX-Tiny (Server CPU)	92.31%	0.581	31.23
EventTube(KITTI)	94.52%	–	33.25
Geometry [10]	92.98%	–	7.15
LOCNet [11]	94.49%	–	6.99

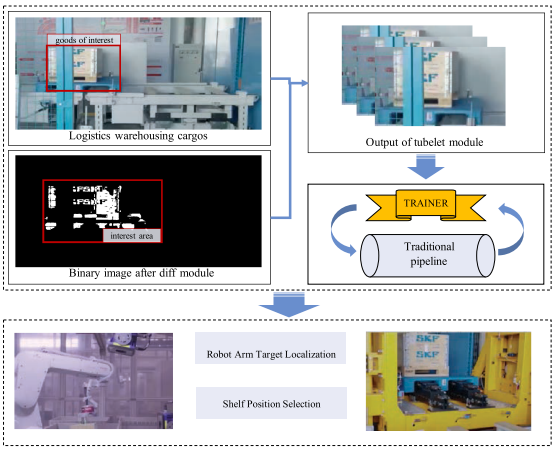


Fig. 7. Application diagram of EventTube in warehousing cargo location and identification.

available monocular camera 3-D detection methods, EventTube can detect 3-D targets faster than conventional 3-D detection methods by compressing the video frame sequences with the filtering of nonevent data.

D. Logistics Applications

There are many AI applications in logistics systems, especially in warehousing, where RGB-Monocular based cargo location and identification is a common application. Since video data usually entails effective communication and computational overhead, real-world field buses are generally incapable of taking up such pressure. Most current research on such problems [21] has focused on resource scheduling and load balancing.

However, optimizing scheduling strategies has not fundamentally reduced the computation and communication overheads in the logistics system, which has failed to reduce logistic costs significantly. This leads to the intelligent logistics system's high efficiency and low cost. Therefore, as shown in Fig. 7, intelligent logistics systems can be combined with edge computing to achieve autopilot of AMRs, localization, and identification of goods, and improved efficiency of logistics and warehousing by using resources on AMRs or terminal devices. Specifically, EventTube uses historical videos captured by cameras as training sets to build neural network models capable of 3-D target detection and localization instead of through manual methods. Then, the diff module is applied to detect the events in the video with high efficiency and hit rate, and the data determined as "Event" is transferred to the tubelet module to filter out the frames that are not useful for machine vision methods without degrading the validity, and to mine the depth information based on consecutive frames for fast 3-D target detection and localization. EventTube can be deployed on AMRs or an edge computing platform, where cameras and storage servers collaborate to fully exploit the computing power of edge devices and improve overall resource utilization. With the help of EventTube, robotic arms, AMRs, and warehousing platforms can quickly and efficiently locate and identify goods and select the corresponding actions and shelf locations.

VI. CONCLUSION

This article introduced EventTube, an edge computing-based event aware system that can efficiently discover events from the video data captured by RGB-Monoculars and collaborate with individual devices to make timely decisions. It proposes an edge computing and accurate and efficient solution for warehouse cargo localization and identification, exploiting the spatio-temporal correlation of events of interest in streaming data to acquire video information on videos captured by monocular cameras quickly. Technical support was provided for the main tasks of logistics scenarios such as AMR environment perception, cargo localization, and 3-D target detection. Extensive experimental results showed the superiority of EventTube in terms of accuracy, time delay, and roll-out cost. Extensive experimental results showed that our EventTube outperforms existing approaches.

For future work, in this article, we will improve the processing capability for high noise video streams by enhancing the layered enhancement method of tubelet and adding support for other industrial video classes so that EventTube can be used effectively in large-scale systems.

REFERENCES

[1] T. Li, B. Huang, C. Li, and M. Huang, "Application of convolution neural network object detection algorithm in logistics warehouse," *J. Eng.*, vol. 2019, no. 23, pp. 9053–9058, 2019.
[2] P. Wang, Y. Wang, X. Wang, Y. Liu, and J. Zhang, "An intelligent actuator of an indoor logistics system based on multi-sensor fusion," in *Actuators*, vol. 10, no. 6, 2021, Art. no. 120.
[3] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on *X*-transformed points," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 828–838.
[4] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. Netravali, "Reducto: On-camera filtering for resource-efficient real-time video analytics," in *Proc. Annu. Conf. ACM Special Int. Group Data Commun. Appl., Technol., Architectures, Protoc. Comput. Commun.*, 2020, pp. 359–376.
[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.
[6] H. Gao, B. Cheng, J. Wang, K. Li, J. Zhao, and D. Li, "Object classification using CNN-based fusion of vision and LIDAR in autonomous vehicle environment," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4224–4231, Sep. 2018.
[7] S. Quan and J. Chen, "AGV localization based on odometry and LiDAR," in *Proc. IEEE 2nd World Conf. Mech. Eng. Intell. Manuf.*, 2019, pp. 483–486.

[8] C.-W. Chen, C.-L. Lin, J.-J. Hsu, S.-P. Tseng, and J.-F. Wang, "Design and implementation of AMR robot based on RGBD, VSLAM and SLAM," in *Proc. IEEE 9th Int. Conf. Orange Technol.*, 2021, pp. 1–5.
[9] L. Zhou et al., "A tightly-coupled positioning system of online calibrated RGB-D camera and wheel odometry based on SE(2) plane constraints," *Electronics*, vol. 10, no. 8, 2021, Art. no. 970.
[10] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká, "3D bounding box estimation using deep learning and geometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5632–5640.
[11] H. Zhang, H. Ji, A. Zheng, J.-N. Hwang, and R.-H. Hwang, "Monocular 3D localization of vehicles in road scenes," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2021, pp. 2855–2864.
[12] Z. Wu, C. Xiong, C.-Y. Ma, R. Socher, and L. S. Davis, "AdaFrame: Adaptive frame selection for fast video recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1278–1287.
[13] C. Zhang, Q. Cao, H. Jiang, W. Zhang, J. Li, and J. Yao, "FFS-VA: A fast filtering system for large-scale video analytics," in *Proc. 47th Int. Conf. Parallel Process.*, 2018, pp. 1–10.
[14] J. Emmons, S. Fouladi, G. Ananthanarayanan, S. Venkataraman, S. Savarese, and K. Winstein, "Cracking open the DNN black-box: Video analytics with DNNs across the camera-cloud boundary," in *Proc. Workshop Hot Topics Video Analytics Intell. Edges*, 2019, pp. 27–32.
[15] K. Du et al., "Server-driven video streaming for deep learning inference," in *Proc. Annu. Conf. ACM Special Int. Group Data Commun. Appl., Technol., Architectures, Protoc. Comput. Commun.*, 2020, pp. 557–570.
[16] J. Chen, X. Liu, Y. Liu, L. Chen, and W. Lu, "The method of AGV detection and cargo status recognition based on globe vision," in *Proc. IEEE 12th Int. Congr. Image Signal Process., BioMed. Eng. Inform.*, 2019, pp. 1–5.
[17] J. Zhang, R. Liu, K. Yin, Z. Wang, M. Gui, and S. Chen, "Intelligent collaborative localization among air-ground robots for industrial environment perception," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9673–9681, Dec. 2019.
[18] F. Niemann, S. Lüdtke, C. Bartelt, and M. Ten Hompel, "Context-aware human activity recognition in industrial processes," *Sensors*, vol. 22, no. 1, 2022, Art. no. 134.
[19] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2022.3179507.
[20] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding YOLO series in 2021," 2021, *arXiv:2107.08430*.
[21] X. Li, J. Wan, H.-N. Dai, M. Imran, M. Xia, and A. Celesti, "A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4225–4234, Jul. 2019.

Yijun Mo (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in communication and information systems from the Huazhong University of Science and Technology, Wuhan, China, in 1999, 2002, and 2008, respectively.

He was a Visiting Scholar with the School of Electronics and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, in 2007. He is currently an Associate Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology. His research interests include edge computing, intelligent network, and cognitive computing. In addition, he is attracted by applying generic techniques to solve and implement real-life demands from different domains.

Zhenchuan Sun received the B.S. degree in computer science from the Dalian University of Technology, Dalian, China, in 2019. He is currently working toward the M.S. degree with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China.

His research interests include edge computing and task offloading.

Chen Yu (Member, IEEE) received the B.S. degree in mathematics and the M.S. degree in computer science from Wuhan University, Wuhan, China, in 1998 and 2002, respectively, and the Ph.D. degree in information science from Tohoku University, Sendai, Japan, in 2005.

From 2005 to 2006, he was a Japan Science and Technology Agency Postdoctoral Researcher with the Japan Advanced Institute of Science and Technology, Nomi, Japan. Since 2008, he has been with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, where he is currently a Professor, working in the areas of ubiquitous computing, edge computing, and edge intelligence.