JITK (JURNAL ILMU PENGETAHUAN DAN TEKNOLOGI KOMPUTER)

VOL. 11. NO. 1 AUGUST 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v11i1.6818

DRIP INFUSION MONITORING AND DATA LOGGING SYSTEM BASED ON YOLOv5

Giri Wahyu Wiriasto[1*]; Andika Rizaldy[1]; Putu Aditya Wiguna[1,2]; Indira Puteri Kinasih[3]

Electrical Engineering Department[1]
Medical and Health Science Department[2]
University of Mataram, Mataram City, Indonesia[1,2]
https://unram.ac.id/en/[1,2]
giriwahyuwiriasto@unram.ac.id*; andika9105@gmail.com; aditya.ku2004@gmail.com

Faculty of Science[3]
Universiti Brunei Darussalam, Bandar Sri Begawan City, Brunei Darussalam[3]
https://ubd.edu.bn/[3]
indiraputeri@ubd.ac.bn

(*) Corresponding Author
(Responsible for the Quality of Paper Content)

Abstract— *Intravenous infusion (IV) functions to deliver medication or fluids directly into the patient's body and requires an accurate drops-per-minute (TPM) calculation to ensure the correct dosage is administered. Manual calculation techniques, which are still widely used today, tend to be inefficient and carry a high risk of human error. Therefore, a more reliable and innovative automated approach is needed. In this study, we developed a prototype of an automatic infusion monitoring system based on the CNN-YOLOv5 architecture. The system records a one-minute IV drip video using a mobile device, then processes it through a server to automatically calculate the TPM, where YOLOv5 is used for drip detection, Deep SORT for object tracking, and a unique ID numbering scheme is applied to each droplet to ensure it is counted only once until it exits the frame. The calculation results are stored in a patient database that we designed. We also explored the effect of dataset background on accuracy. Testing was conducted on 48 videos (30 fps) with two background types—white (LBP) and black (LBH)—and drip variations of 20, 30, 40, and 50 TPM with varying durations. The results showed higher accuracy on the black background, reaching 0.79 compared to 0.58 on the white background, both with a precision of 1.00. The system demonstrated excellent performance in detecting drips with high precision and good accuracy, particularly on LBP for TPM <40 fps and on LBH for TPM <50 fps.*

Keywords: *convolutional neural network, drip infusion, intravenous infusion, YOLOv5*

Abstrak— *Infus intravena (IV) berfungsi untuk mengantarkan obat atau cairan langsung ke dalam tubuh pasien, dan membutuhkan perhitungan tetesan per menit (TPM) yang akurat agar dosis yang diberikan tepat. Teknik perhitungan manual yang masih banyak digunakan saat ini cenderung tidak efisien dan berisiko tinggi terhadap kesalahan manusia. Oleh karena itu, dibutuhkan pendekatan otomatis yang lebih andal dan inovatif. Dalam studi ini, kami mengembangkan prototipe sistem pemantauan infus otomatis berbasis arsitektur CNN-YOLOv5. Sistem merekam video tetesan IV selama satu menit menggunakan perangkat seluler, lalu memprosesnya melalui server untuk menghitung TPM secara otomatis dimana YOLOv5 bekerja untuk deteksi tetesan, Deep SORT untuk pelacakan objek, dan skema penomoran ID unik pada setiap tetesan agar tidak dihitung lebih dari sekali hingga keluar dari bingkai. Hasil penghitungan disimpan dalam basis data pasien yang kami rancang. Kami juga mengeksplorasi efek latar belakang dataset terhadap akurasi. Pengujian dilakukan terhadap 48 video (30 fps) dengan dua jenis latar belakang—putih (LBP) dan hitam (LBH)—dan variasi tetesan 20, 30, 40, serta 50 TPM dengan durasi waktu yang dibedakan. Hasil menunjukkan akurasi lebih tinggi pada latar belakang hitam, mencapai 0.79 dibandingkan 0.58 pada latar belakang putih,*

keduanya dengan presisi 1.00. Sistem ini menunjukkan performa sangat baik dalam mengenali tetesan dengan presisi tinggi, dan akurasi yang baik khususnya pada LBP untuk TPM <40 fps dan LBH untuk TPM <50 fps.

***Kata Kunci*: :** *jaringan syaraf konvolusional, infus tetes, infus intravena, YOLOv5.*

INTRODUCTION

An intravenous (IV) infusion is a medical device used to deliver fluids, medications, or nutrients directly into a patient's bloodstream at regular intervals [1]. This therapy requires medical supervision, particularly to monitor the drip rate to ensure accurate dosage. The fluid flow rate is typically observed through the number of drops in the drip chamber, which is still manually adjusted and counted by medical personnel in most cases [2]. However, this rate may vary due to factors such as fluid volume, the height of the infusion, or blockages in the tubing, necessitating periodic adjustments. Manual counting is time-consuming and prone to errors.

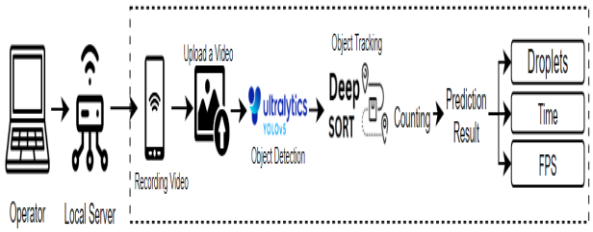
To address this, various studies have proposed automatic infusion monitoring solutions. A system using digital image processing was developed [3], while intravenous fluid delivery was described in [4]. An IoT-based system for continuous monitoring from the nurse's station was introduced [5], and a monitoring system for infused liquid volume via wireless was designed [6].

This research presents a web-based monitoring system for intravenous (IV) infusion therapy that combines YOLOv5 [9][10] object detection with Deep SORT [11] tracking to enable real-time drop counting. While previous computer vision approaches for infusion monitoring have demonstrated the viability of techniques like YOLO and Fast R-CNN [7][8], the current implementation specifically adapts YOLOv5's architecture for clinical web applications, prioritizing usability in inpatient care settings.

The study's preliminary phase employs a controlled evaluation framework using datasets with black (LBH) and white (LBP) backgrounds to isolate the impact of environmental variables. By restricting the object class to only "drops," the investigation aims to establish baseline performance metrics for detection and counting accuracy before expanding to more complex clinical scenarios. This focused approach allows for systematic validation of the core algorithms while maintaining clinical relevance through web-based deployment and real-time processing capabilities that address limitations in existing solutions [7][8] regarding practical implementation in hospital environments. The video dataset was captured under ambient daytime lighting conditions in hospital inpatient rooms, without controlled or additional lighting setups.

MATERIALS AND METHODS

In previous studies, various efforts have been made to monitor intravenous fluids using artificial intelligence to address calculation errors. In this study, a system was developed by integrating an AI model and deploying it on an end-user device. A general illustration of how the system operates via a smartphone is shown in Figure 1, with this research focusing specifically on the elements enclosed in the dashed black box.



Source: (Research Proposed, 2025)

Figure 1. Diagram block of research area

The illustration in Figure 1 shows how users can access the application after the operator starts it and connects to the local network. Connected users can run the application, record videos, and upload them for processing. This process involves object detection using the YOLOv5 algorithm integrated with the Deep SORT algorithm. The output includes information such as the number of drops, time, and frames per second (fps).

Dataset

The dataset used in this study was obtained through direct video recording. A total of 28,800 images were generated by extracting frames from 16 one-minute videos recorded at 30 fps. The data was split into two subsets: 70% for training and 30% for testing. Furthermore, the dataset was divided into two groups based on background color, as shown in Figures 2(a,b) for the white background dataset (LBP) [12] and Figures 2(c,d) for the black background dataset (LBH) [13]. These two datasets were then further light qualitative intensity categorized into four subgroups: Figure 2(a) shows the Bright White Background (TBP), Figure 2(b) shows the Dark White Background (GBP), Figure 2(c) shows the Bright Black Background (TBH), and

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

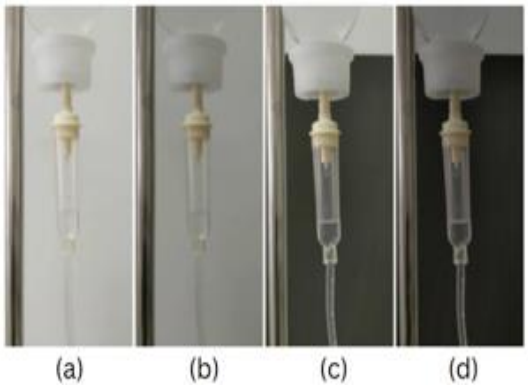
VOL. 11. NO. 1 AUGUST 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v11i1.6818

Figure 2(d) shows the Dark Black Background (GBH). The detailed distribution of each category in the dataset is presented in Table 1.

Table 1. Dataset of video to image drop infusion capture with different sample category

Category	Video .fps to (images)
Bright White Background (TBP)	7200
Dark White Background (GBP)	7200
Bright Black Background (TBH)	7200
Dark Black Background (GBH)	7200
Total	28800

Source: (Research Proposed, 2025)



Source: (Research Proposed, 2025)
Figure 2. Dataset sampling for each category

Labelling Dataset

In the data labeling stage, the process was carried out manually using the Roboflow website [14]. Roboflow is a tool developed by the company Roboflow that facilitates collaborative image annotation and allows for flexible data management, augmentation, and export [15].

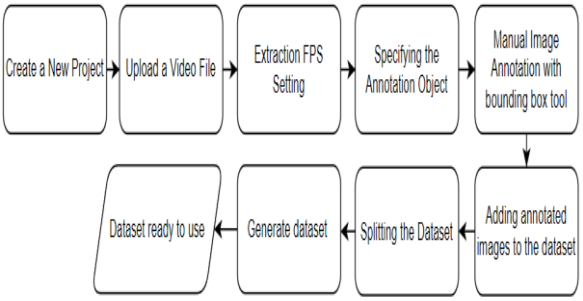
Utilizing a dataset from Roboflow can be done by generating an API after selecting the desired model format. Figure 3 shows the API source code for the dataset that can be used.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="0ksiu0vod9pRsFTZXxr2")
project = rf.workspace("andika-rizaldy").project("set-anak-bp")
dataset = project.version(1).download("yolov5")
```

Source: (Research Result, 2025)
Figure 3. API Dataset Source Code link

Figure 4 explanation the position of the counting baseline from dataset annotation labeling bounding box (BB) objects dimension with annotation tool. The dataset preparation for infusion drip detection begins by creating a new project in Roboflow, where researchers upload high-quality video recordings of IV drips under various clinical conditions. The system first extracts frames at a configurable FPS rate (typically 2-5 FPS) to capture critical droplet formation moments while avoiding redundant data. Each extracted frame undergoes meticulous manual annotation using Roboflow's bounding box tool, where annotators label three key objects: (1) the drip chamber body, (2) individual falling droplets, and (3) the fluid meniscus level. To ensure annotation consistency, the team follows strict guidelines: droplets are marked at their fullest visible diameter, chambers use rectangular boxes encompassing the entire visible column, and ambiguous frames containing bubbles or occlusions are flagged for review. Roboflow then automatically processes the annotated dataset by splitting it into balanced training (70%), validation (20%), and test (10%) sets while applying essential preprocessing (auto-orientation, resizing to 640×640 pixels).



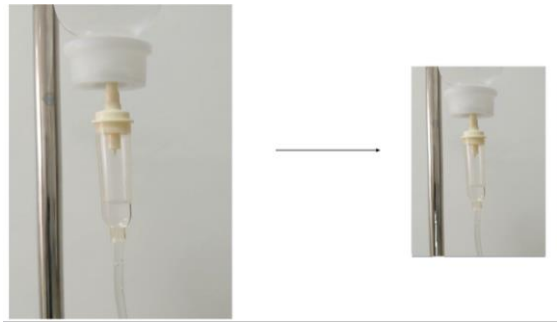
Source: (Research Proposed, 2025)
Figure 4. Block diagram of the roboflow dataset preparations

Pre-processing

In the pre-processing stage, steps are taken to simplify the image format. Various types of pre-processing can be applied, such as resizing images, converting them to grayscale, applying dynamic cropping, and more. In this study, to improve efficiency, input images were resized to a relatively small spatial resolution, from 1920x1080 to 640x640, as implemented in this research. Afterwards, the annotated data was split using a 70:30 ratio, with 70% used for training and 30% for testing the total dataset. The illustration in Figure 5 shows the output of the resized image.

Model Implementation

The model training process was carried out using the PyTorch library. In this study, several hyperparameters were identified, such as batch size, learning rate, and others. The batch size used was 32, with variations in the number of epochs set at 25, 50, and 75. In the experiment with 25 epochs, an accuracy of 0.946 was achieved, with a training time of 25 minutes and 33 seconds. The selection of hyperparameters (batch size, learning rate) and epoch values was performed iteratively using trial-and-error to achieve optimal performance.



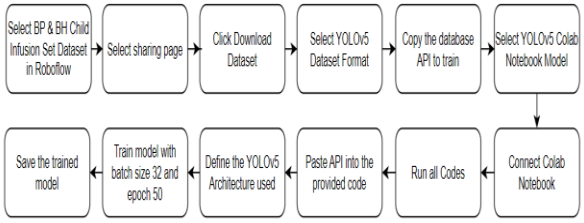
Source: (Research Proposed, 2025)
Figure 5. Resized image dataset sampling

When trained with 50 epochs, the accuracy increased to 0.972, with a training time of 50 minutes and 34 seconds. Meanwhile, in the experiment with 75 epochs, the accuracy reached 0.979, and the training time was 1 hour, 10 minutes, and 32 seconds.

Figure 6 illustrates the stages of the model training process. The infusion drip detection model training process begins by selecting the BP (image) with white background) & BH (image with black background) image dataset from Roboflow, a specialized platform for computer vision datasets. Through Roboflow's sharing portal, users initiate the dataset export by clicking the "Download Dataset" button and selecting the YOLOv5-compatible format, which structures the data with images in a dedicated folder and corresponding text annotations for bounding boxes.

The platform generates a unique API endpoint containing dataset version information and preprocessing settings, which is then copied for integration into the training pipeline. Researchers connect this dataset to a YOLOv5 implementation within a Google Colab notebook, first cloning the official Ultralytics repository and installing necessary dependencies. Critical training parameters are configured, including a batch size of 32 optimized for GPU memory usage, 50 training epochs for balanced convergence, and a 640x640 pixel input resolution. The YOLOv5 architecture variant (typically selecting among nano, small, medium, or large models) is specified based on performance requirements.

The Roboflow API is integrated into the training script using cURL commands for dataset downloading and extraction. When executed, the notebook sequentially processes data preparation, model initialization, the training loop with forward-backward propagation, and validation against test sets. The final optimized model weights are saved as best.pt, containing not only the trained parameters but also architectural details, class names, and training metadata for future deployment in real-time infusion monitoring systems. This end-to-end workflow ensures proper configuration of all components from data sourcing to model output while maintaining reproducibility through version-controlled datasets and standardized YOLOv5 implementations.



Source: (Research Proposed, 2025)
Figure 6. Block Diagram of Model Training

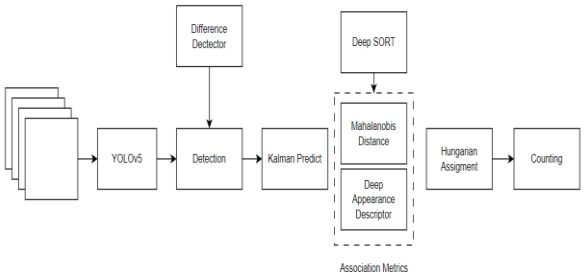
The saved model is then integrated with the YOLOv5 algorithm, Deep SORT, and a custom-developed counting algorithm. The entire model training process is outlined in the illustration in Figure 7, which includes the workflow of all the algorithms involved. The input video is first processed using the YOLOv5 algorithm for object detection, and the detection results are then passed to the tracking stage using the Deep SORT algorithm before the objects are counted.

YOLOv5 Architecture

You Only Look Once (YOLO) is a well-known object detection method recognized for its speed. This method uses a single convolutional network to predict objects within an image. Its efficiency is achieved by dividing the image into cells or grids, with each cell responsible for predicting a number of bounding boxes (BB), the confidence score for each box, and the class probabilities.

During testing, YOLO multiplies the conditional class probabilities with the individual box confidence predictions, as shown in Equation (1).

$$Pr(Class_i|Object) * Pr(Object) * IoU_{Pred}^{Truth} = Pr(Class_i) * IoU_{Pred}^{Truth} \quad (1)$$



Source: (Research Proposed (from [11][19], 2025)
Figure 7. YOLO and Deep SORT Cross-section Exposure

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 11. NO. 1 AUGUST 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v11i1.6818

Equation (1) consists of the conditional class probability ($Pr(Class_i|Object)$), the confidence score of the predicted box ($Pr(Object) * IoU_{Pred}^{Truth}$), and the class confidence score ($Pr(Class_i) * IoU_{Pred}^{Truth}$). The YOLOv5 algorithm adopts a CNN-based object detection approach. Convolutional Neural Networks (CNNs) are a type of neural network architecture highly effective at recognizing patterns and features in image data. YOLOv5 employs EfficientNet as the foundation of its model architecture [16]. EfficientNet is a powerful and efficient CNN architecture for image recognition. The YOLOv5 structure is divided into three main components: (1) a backbone using CSPDarknet, (2) a neck using PANet, and (3) a head that consists of the YOLO layer.

The process begins with input data fed into CSPDarknet to extract features, which are then integrated using PANet to aggregate multi-scale features. Finally, the YOLO layer produces the detection results, which include class information, confidence scores, object location, and size.

In the YOLOv5 architecture, the convolutional layers consist of the following components: the input image, a Convolution-Downsampling Block to extract features from the image, a Dense Connection Block to enhance information flow and feature representation, a Spatial Pyramid Pooling Block to merge multi-scale features, and an Object Detection Block to predict bounding boxes and class probabilities for objects in the image. The input image is divided into a grid of small cells, with each cell responsible for detecting potential objects within it. The CNN processes the entire image as a whole and outputs probability predictions for object classes along with bounding boxes surrounding the detected objects [17].

Deep SORT

The video input successfully detected by YOLOv5 is then assigned a unique ID, allowing the same object to be consistently counted as a single entity using an object tracking algorithm [18]. Object tracking involves several key components when using Deep SORT [19], including the Kalman Filter (KF), Association Metrics (AM), and Hungarian Assignment (HA).

The A Kalman Filter serves as a predictive algorithm in a tracking system. It works by estimating an object's position in the next video frame. Once the object is detected, the filter compares this prediction to the actual position, then updates its prediction for subsequent frames. This process ensures the tracking remains stable and accurate, even if the object moves quickly or is temporarily obstructed, because the filter continuously corrects its estimates. To identify the same object across different frames, Association Metrics like the Mahalanobis Distance (MD) and Deep Appearance Descriptor (DAD) are used. MD measures how close a new detection is to the predicted position, taking uncertainty into account. Meanwhile, DAD acts as the object's visual origin data, comparing the visual appearance of a new detection to that of a tracked object. By combining the positional prediction from the Kalman Filter with identity verification from the MD and DAD, Deep SORT can track objects precisely and reliably, even in complex situations [19].

Finally, the object detection-to-tracking association problem is resolved using the Hungarian Assignment algorithm, which maps detected objects to the most appropriate tracks. The Hungarian algorithm minimizes the total distance between detections and tracks, ensuring an optimal mapping needed to update the object tracking effectively.

$$d(D_i, P_i) = 1 - \frac{\sqrt{(u_{D_i} - u_{P_i})^2 + (v_{D_i} - v_{P_i})^2}}{\frac{1}{2}\sqrt{h^2 + w^2}} \quad (2)$$

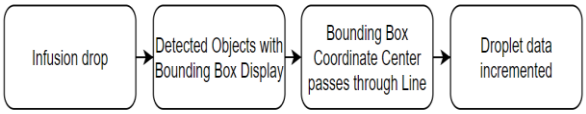
In the equation (2), (h, w) represents the height and width of the input image, D_i is the bounding box from the detection set that contains positional components (u_{D_i}, v_{D_i}), and P_i is the bounding box from the prediction set that contains positional components (u_{P_i}, v_{P_i}).

Drip Detect-counting of Diagram Block

The calculation of IV drip objects is performed by creating a dedicated function. The counting process begins when the center point of the object's bounding box (BB) crosses a predefined line [20]. The flow of this counting process is illustrated in the Flowchart shown in Figure 8. As a drop falls, the object is detected, and its bounding box is displayed. Counting occurs when the center of the object's BB crosses the designated line, ensuring that the same object is not counted more than once.

Model Evaluation

To evaluate the model's performance, standard metrics such as the confusion matrix, accuracy, precision, recall, and F1-score are used. These performance indicators are based on the methodology employed by Markoulidakis,.et.al[21].



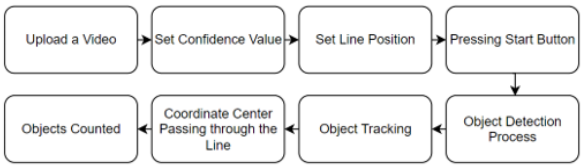
Source: (Research Proposed, 2025)

Figure 8. Diagram Blok Bounding Box for Increment Counted

System Design

As part of the implementation phase, the author developed a web-based application. This web application was built using the Streamlit framework with Python on a personal laptop running Windows 11, equipped with an AMD Ryzen 6600H processor, 16 GB of RAM, and an AMD Radeon graphics card. Testing was conducted throughout the development process.

Once the YOLOv5 model was trained and tested to achieve the desired results, it was saved in the .pt format using the PyTorch library [22]. PyTorch is a machine learning framework based on the Torch library, developed by Meta AI for computer vision [23] and natural language processing [24] applications.

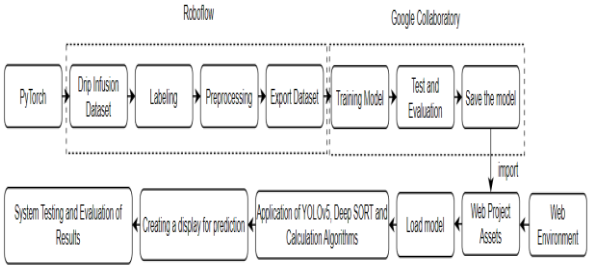


Source: (Research Proposed, 2025)

Figure 9. System Workflow Block Diagram

The system workflow is illustrated in the block diagram shown in Figure 9. Users upload a video to be analyzed, then set the confidence threshold and define the line position for counting. After pressing the start button, the process begins with object prediction followed by object tracking. When the center coordinate of the object's bounding box (BB) crosses the predefined line, the object is counted.

System Integration



Source: (Research Proposed, 2025)

Figure 10. Integration model to Web-based of Patient management system

Figure 10 illustrates the sequence of steps involved in integrating the model into the web application. After acquiring the dataset, the labeling process is carried out, followed by preprocessing. The model is then developed based on the proposed initial design. It undergoes training, testing, and evaluation until it reaches satisfactory performance. To enable the model's use on other devices, such as laptops, *PyTorch* provides a library to convert the trained model into a .pt format. This file is then imported into the web project directory.

In the web application design process, the next step involves creating a prediction function that processes data using the imported model. The uploaded video data is passed through the prediction stage, where the model generates predictions. These predictions determine the final count of intravenous fluid drops.

Software testing result

After integrating the model into the designed system, software testing was conducted to evaluate its performance using a black-box testing approach. The evaluation involved a dataset of 48 videos with varying drip rates: 20 DPM, 30 DPM, 40 DPM, and 50 DPM. Each drip rate was tested with three different durations: 10 seconds, 15 seconds, and 60 seconds. The testing was divided into four categories, where the combination of TBP and GBP represents the LBP group, and TBH and GBH represent the LBH group.

RESULTS AND DISCUSSION

Testing was carried out using 48 videos recorded at 30 frames per second (FPS). In one scenario—using a drip rate of 20 drops per minute (DPM), a 10-second duration, and a confidence threshold of 0.75—each droplet was detected across a variable number of frames. For example, the first three droplets lasted 25, 22, and 24 frames, respectively, with intervals of approximately 60 to 70 frames between them. These results indicate that the system can detect droplets with consistent frame durations, suggesting adequate temporal sensitivity of the model to visual changes. Further test results are detailed in Tables 4 and 5.

Table 4. Test Results with LBP Dataset

Fps Dataset	Actual		Predict		Total images frame	
	TBP	GBP	TBP	GBP	TBP	GBP
20 TPM (10s)	3	3	3	3	71	75
20 TPM (15s)	4	4	4	4	100	102
20 TPM (60s)	16	18	16	18	400	450
30 TPM (10s)	3	4	3	4	66	88
30 TPM (15s)	7	6	7	6	154	138
30 TPM (60s)	26	25	24	23	528	508
40 TPM (10s)	5	6	5	6	90	110
40 TPM (15s)	8	8	6	8	108	146
40 TPM (60s)	32	32	29	27	522	488
50 TPM (10s)	7	6	6	6	86	88
50 TPM (15s)	10	9	8	9	112	127
50 TPM (60s)	41	42	35	23	490	324

Source: (Research Result, 2025)

Accredited Rank 2 (Sinta 2) based on the Decree of the Dirjen Penguatan RisBang Kemenristekdikti No.225/E/KPT/2022, December 07, 2022. Published by LPPM Universitas Nusa Mandiri

VOL. 11. NO. 1 AUGUST 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v11i1.6818

Based on the data in Tables 4 and 5, the prediction results appear to be fairly accurate for durations of 10 seconds and 15 seconds. However, for videos with durations longer than 30 seconds, the counted number of drops tends to deviate from the actual number.

Table 5. Test Results Using the LBH Dataset

Fps Dataset	Actual		Predict		Total image Frame	
	TBH	GBH	TBH	GBH	TBH	GBH
20 TPM (10s)	3	5	3	5	75	125
20 TPM (15s)	5	6	5	6	125	150
20 TPM (60s)	17	18	17	18	425	450
30 TPM (10s)	5	5	5	5	110	110
30 TPM (15s)	7	7	7	7	154	154
30 TPM (60s)	26	26	25	26	550	572
40 TPM (10s)	6	6	6	6	108	108
40 TPM (15s)	8	9	8	9	144	162
40 TPM (60s)	32	34	32	33	576	594
50 TPM (10s)	7	7	7	7	98	98
50 TPM (15s)	8	11	8	10	112	140
50 TPM (60s)	39	42	38	37	532	518

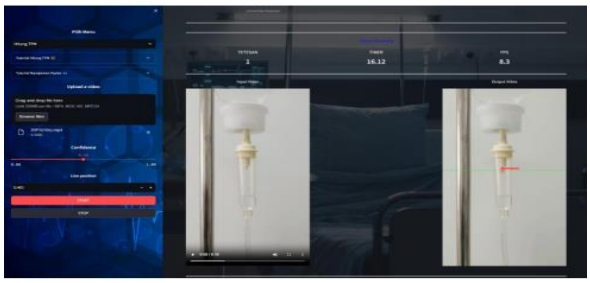
Source: (Research Result, 2025)

This may be due to the very short interval between one droplet and the next, causing the system to interpret multiple droplets as a single object. Videos using the LBH dataset produced more accurate counting results because the droplets were more clearly visible, unlike in the LBP dataset, where the background color closely resembles the color of the droplets. With high precision and good accuracy, the system effectively detected drips. The test results showed the best performance on a white background (LBP) with a drip rate below 40 TPM, and on a black background (LBH) with a drip rate below 50 TPM.

Source: (Research Result, 2025)
Figure 11. Web-Based Application Interface Design

Web-based Application Interface for Patient TPM Data Management

This web application is a direct implementation of the previously trained model, enabling real-time counting of intravenous (IV) fluid drops. The main structure of the application is divided into two key sections: the drop count calculation page and the patient management page. Navigation between these pages is facilitated through the sidebar menu. The application interface is shown in Figure 11. As shown in figure 12, The page appears once the prediction process is complete. It will show the prediction results generated by the user. The page also includes additional information such as status, drip count, time, and .fps.



Source: (Research Result, 2025)
Figure 12. Web-Based Application Interface Design

Confusion Matrix

Based on the number of test data, a total of 48 videos recorded at 30 FPS were divided into two groups: 24 for LBP and 24 LBH. The confusion matrix categorizes the data into four groups: data that is correctly predicted falls under True Positive (TP), data with a predicted count higher than the actual value is categorized as False Positive (FP), data with a predicted count lower than the actual value is classified as False Negative (FN), and data that is not detected in both the prediction and actual counts falls under True Negative (TN). The testing was conducted using a threshold of 0.75 for each data testing. The confusion matrix for each background type can be found in Table 6.

Table 2. Confusion Matrix

Parameter	LBP	LBH
Accuracy	0.58	0.79
Precision	1.00	1.00
Recall	0.58	0.79
F1-Score	0.74	0.88

Source: (Research Result, 2025)

Based on the data in Table 6, it can be concluded that the identification performance on test data with a LBH is superior to that with a LBP. The identification accuracy with LBH reaches 0.79, while with LBP it only reaches 0.58. However, in terms of precision, the drip objects were detected accurately with no errors out of a total of 24 test data points. The results were as follows: LBP with 14 TP and 10 FN and LBH with 19 TP and 5 FN.

CONCLUSION

The results of this study show that a real-time, web-based drip infusion monitoring system using the YOLOv5 and DeepSORT algorithms has been developed and integrated with a patient database to serve as historical data. The system is capable of detecting and counting the number of falling IV droplets. Model training was conducted using a batch size of 32 with varying numbers of epochs: 25, 50, and 75. In the experiment with 25 epochs, the model achieved an accuracy of 0.946 with a training time of 25 minutes and 33 seconds. At 50 epochs, the accuracy increased to 0.972, with a training time of 50 minutes and 34 seconds. At 75 epochs, the accuracy reached 0.979 with a training time of 1 hour, 10 minutes, and 32 seconds.

Testing on the combined test dataset (LBP + LBH), comprising 48 videos, showed an integrated performance accuracy of 0.58 for the LBP background and 0.79 for the LBH background. The prediction results were reasonably accurate for video durations of 10 and 15 seconds. However, for videos longer than 30 seconds, the number of counted droplets tended to deviate from the actual number. This may be due to very short intervals between droplets, causing the system to interpret multiple drops as a single object. The achievements of this study contribute to the development of real-time medical monitoring technology in inpatient care settings, particularly regarding the use of suitable IV fluid containers—such as LBH—which support enhanced detection performance. Future research is expected to further improve the system's ability to count infusion droplets more accurately..

REFERENCE

[1] Open Resources for Nursing (Open RN); Ernstmeyer K, Christman E, editors. Nursing Skills [Internet]. Eau Claire (WI): Chippewa Valley Technical College; 2021. Chapter 23 IV Therapy Management. Available from: https://www.ncbi.nlm.nih.gov/books/NBK593209/

[2] K. Venkatesh, S. S. Alagundagi, V. Garg, K. Pasala, D. Karia, and M. Arora, "DripOMeter: An open-source opto-electronic system for intravenous (IV) infusion monitoring," *HardwareX*, vol. 12, p. e00345, 2022, doi: 10.1016/j.ohx.2022.e00345.

[3] S. Song, S. Yan, S. Zhang, and Y. Jiang, "Design of an infusion monitoring system based on image processing," *J. Phys.: Conf. Ser.*, vol. 2037, no. 1, p. 012109, Sep. 2021, doi: 10.1088/1742-6596/2037/1/012109.

[4] Karen Kan, Wilton C. Levine,16 - Infusion Pumps,Editor(s): Jan Ehrenwerth, James B. Eisenkraft, James M. Berry, Anesthesia Equipment (Third Edition), W.B. Saunders, 2021, Pages 351-367, ISBN 9780323672795, https://doi.org/10.1016/B978-0-323-67279-5.00016-9.

[5] S. A. Kadiran, E. Supriyanto, and M. Y. Maghribi, "Sistem Monitoring dan Controlling Cairan Infus Berbasis Website," *J. Riset Rekayasa Elektro*, vol. 5, no. 1, 2023, doi: 10.30595/jrre.v5i1.17743.

[6] M. Z. Samsono Hadi, H. Mahmudah and L. Thania, "Design of Monitoring System for Infused Liquid Volume Based Wireless Communication," *2021 International Conference on Computer Science and Engineering (IC2SE)*, Padang, Indonesia, 2021, pp. 1-6, doi: 10.1109/IC2SE52832.2021.9792048.

[7] N. Giaquinto, M. Scarpetta, M. A. Ragolia, and P. Pappalardi, "Real-time drip infusion monitoring through a computer vision system," in *IEEE Med. Meas. Appl. (MeMeA)*, 2020, doi: 10.1109/MeMeA49120.2020.9137359.

[8] N. Giaquinto, M. Scarpetta, M. Spadavecchia, and G. Andria, "Deep learning-based computer vision for real-time intravenous drip infusion monitoring," *IEEE Sens. J.*, vol. 21, no. 13, 2021, doi: 10.1109/JSEN.2020.3039009.

[9] Ultralytics, "YOLOv5," GitHub. [Online]. Available: https://github.com/ultralytics/yolov5 [Accessed: Jul. 17, 2025].

[10] U. Nepal and H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs," *Sensors*, vol. 22, no. 2, 2022, doi: 10.3390/s22020464.

[11] B. Gao, "Research on Two-Way Detection of YOLO V5s+Deep Sort Road Vehicles Based on Attention Mechanism," in *J. Phys.: Conf. Ser.*, vol. 2303, 2022, doi: 10.1088/1742-6596/2303/1/012057.

[12] A. Rizaldy, "Set Anak Background Putih," roboflow.com. Accessed: Jan. 16, 2024. [Online]. Available: https://universe.roboflow.com/andika-rizaldy/set-anak-bp

[13] A. Rizaldy, "Set Anak Background Hitam," roboflow.com. Accessed: Jan. 16, 2024. [Online]. Available: https://universe.roboflow.com/andika-rizaldy/set-anak-bh

[14] Q. Lin, G. Ye, J. Wang, and H. Liu, "RoboFlow: a Data-centric Workflow Management

VOL. 11. NO. 1 AUGUST 2025
P-ISSN: 2685-8223 | E-ISSN: 2527-4864
DOI: 10.33480/jitk.v11i1.6818

System for Developing AI-enhanced Robots," in *Proc. Mach. Learn. Res.*, 2021.

[15] M. A. Barayan *et al*., "Effectiveness of Machine Learning in Assessing the Diagnostic Quality of Bitewing Radiographs," *Appl. Sci. (Switzerland)*, vol. 12, no. 19, 2022, doi: 10.3390/app12199588.

[16] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, "A forest fire detection system based on ensemble learning," *Forests*, vol. 12, no. 2, 2021, doi: 10.3390/f12020217.

[17] D. Permana and J. Sutopo, "Aplikasi Pengenalan Abjad Sistem Isyarat Bahasa Indonesia (SIBI) Dengan Algoritma YOLOv5," *J. Inform.*, vol. 11, no. 2, 2023.

[18] R. Pereira, G. Carvalho, L. Garrote, and U. J. Nunes, "Sort and Deep-SORT Based Multi-Object Tracking for Mobile Robotics: Evaluation with New Data Association Metrics," *Appl. Sci. (Switzerland)*, vol. 12, no. 3, 2022, doi: 10.3390/app12031319.

[19] Abed, Almustafa & Akrout, Belhassen & Amous, Ikram. "Deep learning-based few-shot person re-identification from top-view RGB and depth images. Neural Computing and Applications". 36. 19365-19382. 2024, doi:10.1007/s00521-024-10239-6.

[20] Thien, "Vehicle Detection and Counting System on Streamlit." *GitHub*, Mar. 25, 2023. Accessed: Aug. 18, 2023. [Online]. Available: https://github.com/npq-thien/Vehicle_Detection_and_Counting_System/activity?activity_type=direct_push

[21] I. Markoulidakis, I. Rallis, I. Georgoulas, G. Kopsiaftis, A. Doulamis, and N. Doulamis, "Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem," *Technologies (Basel)*, vol. 9, no. 4, 2021, doi: 10.3390/technologies9040081.

[22] O. C. Novac *et al*., "Analysis of the Application Efficiency of TensorFlow and PyTorch in Convolutional Neural Network," *Sensors*, vol. 22, no. 22, 2022, doi: 10.3390/s22228872.

[23] A. A. Khan, A. A. Laghari, and S. A. Awan, "Machine Learning in Computer Vision: A Review," *EAI Endorsed Trans. Scalable Inf. Syst.*, vol. 8, no. 32, 2021, doi: 10.4108/eai.21-4-2021.169418.

[24] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: state of the art, current trends and challenges," *Multimed Tools Appl.*, vol. 82, no. 3, 2023 , doi: 10.1007/s11042-022-13428-4.