Real-Time Object Detection for Smart Connected Worker in 3D Printing

Shijie Bian[1,3] (Presenter), Tiancheng Lin[1], Chen Li[1,3], Yongwei Fu[1,2], Mengrui Jiang[1], Tongzi Wu[1], Xiyi Hang[4], **Bingbing Li[1,2] (Corresponding Author)**

1. **Autonomy Research Center for STEAHM, California State University, Northridge, USA**
2. Department of Manufacturing Systems Engineering and Management, California State University, Northridge, USA
3. **Department of Mathematics, University of California, Los Angeles, USA**
4. Department of Electrical and Computer Engineering, California State University, Northridge, USA

TABLE OF CONTENTS

01

Introduction

Background, Motivation, and Contribution

Background, Motivation, and Contribution

Background

1. Rapid advancement in the fields of machine learning and computer vision
2. Stronger GPUs designed for parallel computing allowed powerful algorithms to be developed (e.g., YOLO)

Motivation

1. Main goal: **Offer affordable and scalable smart manufacturing capacities to small and medium-sized manufacturers (SMMs)** – to deal with the different proprietary interfaces as well as communication protocols of various machines
2. Instead of relying on labor-intensive monitoring or the feedback from machines themselves, **could we use computer vision-based real-time monitoring**?

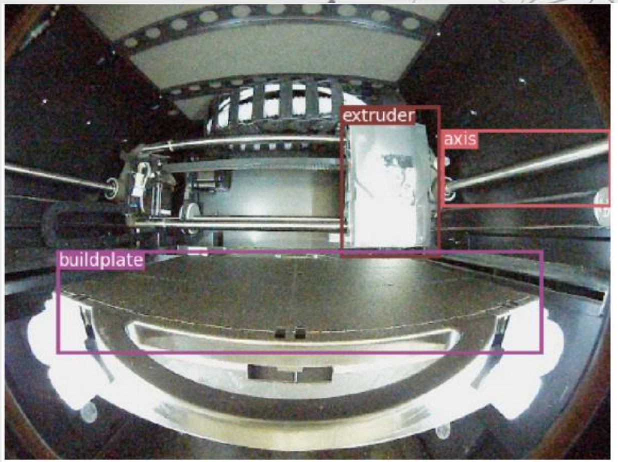
Contribution

1. YOLO-based object detection model to identify positions of machine components
2. A filtering algorithm with look-back functionalities to enhance robustness and make machine state predictions from previous outputs
3. An encapsulated and fully automated monitoring system with a GUI

02
Background

3D Printer Monitoring
Object Detection Algorithms

3D Printer State Monitoring



1. Initialized State

Printer starts and processes the printing command(s)

2. Testing State

All components ready their positions

3. Calibration State

The extruder calibrates its location and finds the starting position

The goal is to predict the 3D printer's machine state in real-time through the position of three essential interior components by analyzing the interior camera image via an object detection algorithm.

4. Heating State

The nozzle and chamber heat up

5. Printing State

The extruder ejects the model and support materials onto the build plate

6. Ending State

All components are reset to their original position

Object Detection Algorithms

Region Based Convolutional Neural Networks (R-CNN)

1. Combines regional selective search with the Convolutional Neural Network (CNN) model

2. Input image segmented into small regions and are combined based on certain features

3. CNN applied to extract features for classification

Example: Mask-RCNN (2018)

Single-Shot Detectors (SSD)

1. Takes only one shot to detect multiple objects

2. Input images with pre-defined bounding boxes

3. Base neural network using VGG-16 for feature extraction

4. Intersection over union (IoU) for locating anchor boxes closest to ground truth

Example: YOLO (2018)

03

Methodology

YOLO-based Object Detection Model
Machine State Filtering Algorithm

Overall Workflow

YOLO-based Object Detection

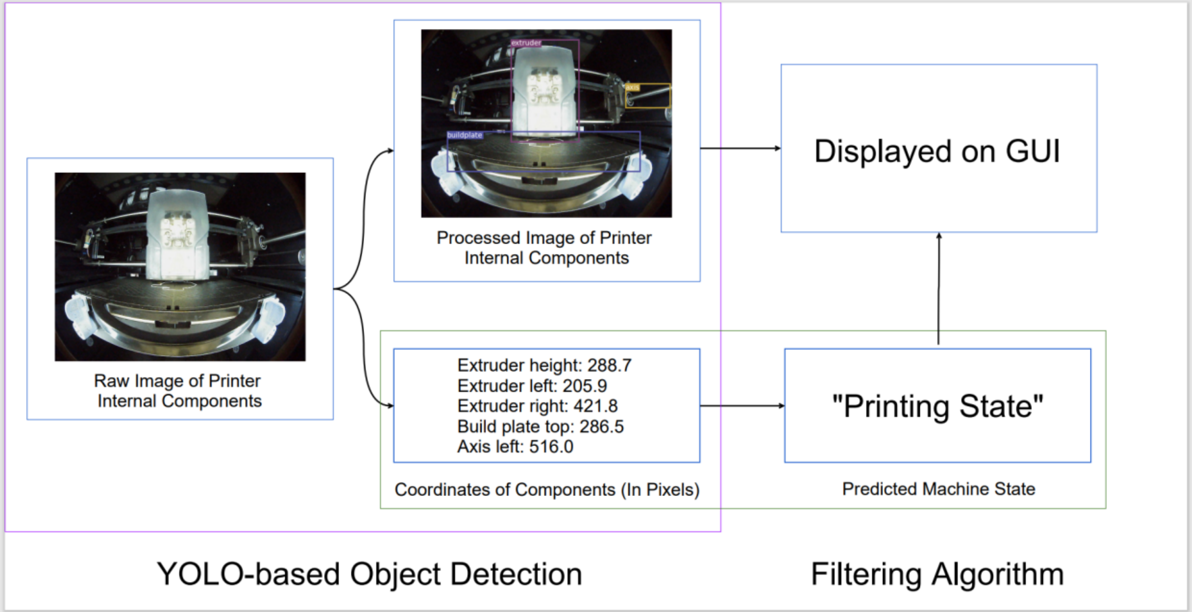
For real-time identification of the relative coordinates of the major components of a 3D printer

Filtering Algorithm

Based on the output from previous module, filter out the most possible machine state from all possible candidates

Display on GUI

Display the results (processed image and predicted machine states) on a webpage-based GUI



A YOLO-based Object Detection Model

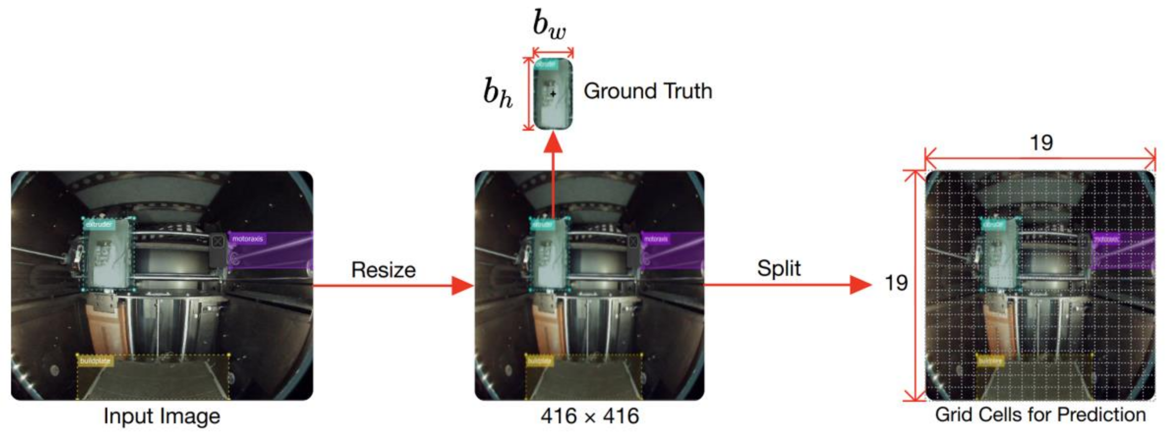


Fig. 2. Illustration of the preprocessing stage

1. Input: digital image, pre-determined class labels (one-hot encoding), bounding boxes – ground truth
2. Bounding boxes: defined by the center x-coordinate, the center y-coordinate, the width, and the height
3. Resized into 416 x 416 (pixels) for standardization
4. Input image split to 19 x 19 grid cells, each of which contain 3 pre-defined anchor bounding boxes
5. Information fed into the deep CNN structure of YOLO V-3, and feature maps are obtained
6. For each of the 3 anchor boxes, the one closest to the ground truth is filtered out using IoU
7. Output contains the processed image, and the coordinates of the bounding boxes for each machine component

GitHub: https://github.com/BrandonBian/SCW-object-detection

Filtering Algorithm for Machine States

Initialized State: The extruder is at the left-back corner, while the build plate is at the bottom;
Testing State: The extruder remains at the same position, the build plate elevates to the top;
Calibration State: The extruder moves forward, while the build plate remains at the same position;
Heating State: The extruder returns to the left back corner, while the build plate remains at the same position;
Printing State: The extruder moves at the front positions, while the build plate gradually descends;
Ending State: The extruder returns to the left back corner, while the build plate descends to the bottom.

1. Input: the coordinates of the machine components acquired from pervious module
2. Constantly checks and stores the current machine state
3. Check whether the movement of the machine components indicate a change in state
4. For each time-stamp, look back at the past 5 frames to eliminate inaccuracy caused by missed object detections
5. Can be used to record machine state transition logs, and for fault detection

GitHub: https://github.com/BrandonBian/SCW-object-detection

04

Experimental Evaluation

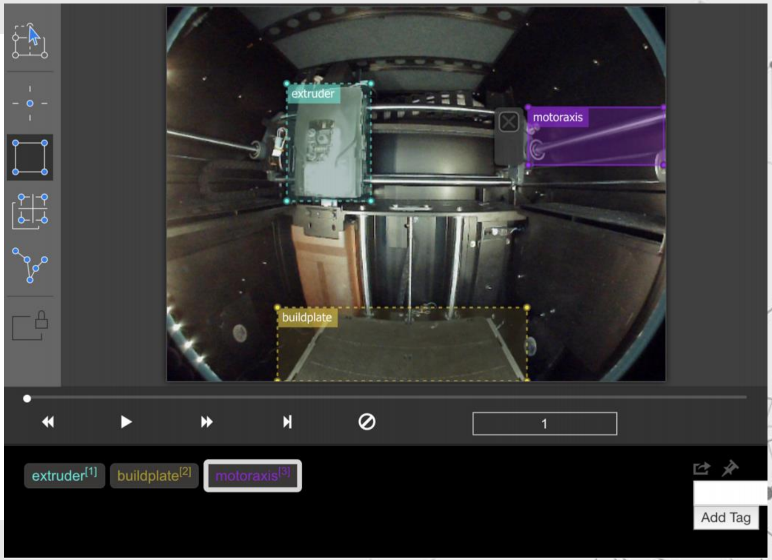
Configurations, Results, and Discussions

Configurations

1. 3D Printer: Stratasys uPrint SE 3D Printer
2. Camera: SVPRO Fisheye Lens 180° USB heat-resistant camera with resolution 1080P and frame rate 30 frames per second (fps)

Dataset Collection

1. Printing of a sample model (2300 image frames)
2. Manual labeling of the image frames (YOLO Visual Object Tagging Tool (VoTT))
3. 80% Training, 20% Validation

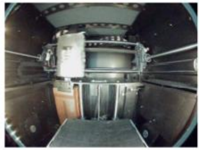
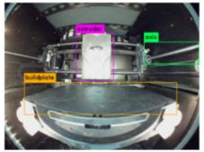
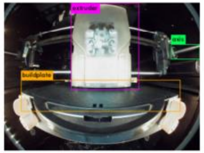
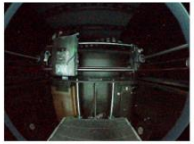
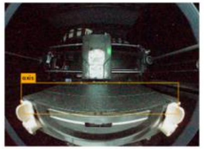
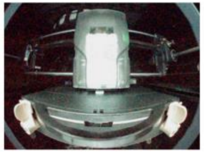


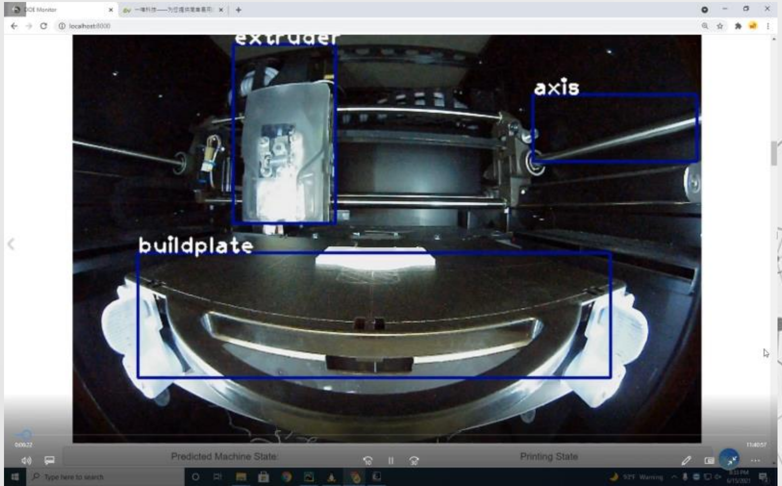
Results

1. During training and validation: 3 hours' training, average class detection accuracy of 0.999, average IoU of 0.922

2. During testing with a sample video: average prediction accuracy of 94.8%, average confidence of 0.87, average inference speed of 0.01 seconds per frame

3. Machine state prediction: average accuracy of 100%

	Training data	Detection result comparison 1	Detection result Comparison 2	Training performance and result:
Light environment training				Converged in 10000 epoches IoU: 0.922
Dark environment training				Converged in 40000 epochs IoU: 0.702



	Feedback Accuracy	Inference Time (s)
YOLO-V3	94.8 %	0.01 s (100 FPS)
Tera Term	100 %	3.81 s (0.26)
Mask-RCNN	95.8 %	0.17 s (6 FPS)

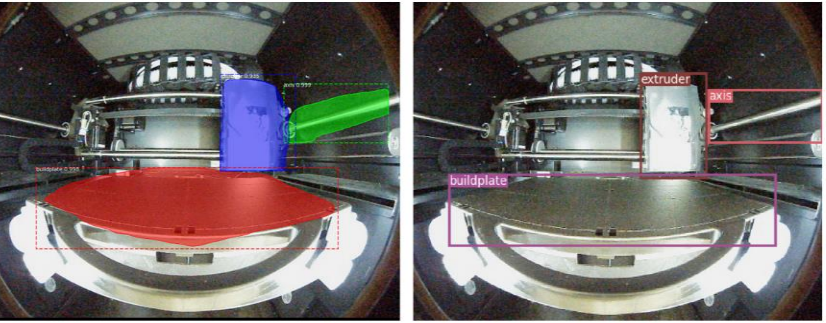


Fig. 7. Test results of Mask R-CNN (left) and the proposed model (right)

Discussion

1. Mask-RCNN: slightly better accuracy, but significantly slower speed

2. Tera Term: perfect accuracy, but longer delay, less scalability and flexibility

Considering the main goal of achieving a real-time system, and the presence of the filtering algorithm that can enhance the robustness of machine state identification, the optimization of response time and inference speed should be prioritized

05

Conclusion

Conclusion and Future Efforts

Conclusion and Future Efforts

Conclusion

1. A YOLO-based object detection model for real-time identification of the positions of critical machine components

2. A filtering algorithm for predicting the machine states

3. The possibility of adopting computer vision algorithms for advanced manufacturing systems

Future Efforts

1. Training with more diverse dataset to enhance the prediction accuracy of the model

2. Detecting machine states of more sophisticated machinery

3. Building an automated and centralized Smart Connected Worker for SMMs

Thank you!

Open Discussion and Q & A

CREDITS