The International Journal of Advanced Manufacturing Technology (2023) 126:1397–1416
https://doi.org/10.1007/s00170-023-11151-4

ORIGINAL ARTICLE

Ladle pouring process parameter and quality estimation using Mask R-CNN and contrast-limited adaptive histogram equalisation

Callum O'Donovan[1] · Ivan Popov[1] · Grazia Todeschini[1,2] · Cinzia Giannetti[1]

Received: 28 August 2022 / Accepted: 23 February 2023 / Published online: 11 March 2023

Abstract
Deep learning in computer vision is becoming increasingly popular and useful for tracking object movement in many application areas, due to data collection burgeoning from the rise of the Internet of Things (IoT) and Big Data. So far, computer vision has been used in industry predominantly for quality inspection purposes such as surface defect detection; however, an emergent research area is the application for process monitoring involving tracking moving machinery in real time. In steelmaking, the deployment of computer vision for process monitoring is hindered by harsh environments, poor lighting conditions and fume presence. Therefore, application of computer vision remains unplumbed. This paper proposes a novel method for tracking hot metal ladles during pouring in poor lighting. The proposed method uses contrast-limited adaptive histogram equalisation (CLAHE) for contrast enhancement, Mask R-CNN for segmentation prediction and Kalman filters for improving predictions. Pixel-level tracking enables pouring height and rotation angle estimation which are controllable parameters. Flame severity is also estimated to indicate process quality. The method has been validated with real data collected from ladle pours. Currently, no publications presenting a method for tracking ladle pours exist. The model achieved a mean average precision (mAP) of 0.61 by the Microsoft Common Objects in Context (MSCOCO) standard. It measures key process parameters and process quality in processes with high variability, which significantly contributes to process enhancement through root-cause analysis, process optimisation and predictive maintenance. With real-time tracking, predictions could automate ladle controls for closed-loop control to minimise emissions and eliminate variability from human error.

Keywords Computer vision · Deep learning · Manufacturing · Remote condition monitoring · Segmentation

Ivan Popov, Dr. Grazia Todeschini and Prof. Cinzia Giannetti contributed equally to this work.

✉ Callum O'Donovan
833887@swansea.ac.uk

Ivan Popov
708967@swansea.ac.uk

Grazia Todeschini
grazia.todeschini@kcl.ac.uk

Cinzia Giannetti
c.giannetti@swansea.ac.uk

[1] Faculty of Science and Engineering, Swansea University, Fabian Way, Swansea SA1 8EN, Wales

[2] King's College London, Strand Campus, London WC2R 2LS, England

1 Introduction

In a conventional Basic Oxygen Steel (BOS) plant, hot metal (HM) ladles are typically utilised for movement, pouring and short-term refractory storage of up to several hundred tonnes of liquid iron and steel. Normally, HM ladles consist of a metallic shell approximately 6 m in height and 5 m in width. This is lined with refractory material, designed to protect the metallic shell and prolong the useful life. Ladles are transported by heavy duty gantry cranes that lift and elevate via lifting hooks [1]. Liquid metal is poured inside Basic Oxygen Furnace (BOF) vessels (shown in Fig. 1), where liquid iron and metal scrap are turned into steel through an exothermic reaction by blowing oxygen into the melt at supersonic speeds. The vessels are tilted for easier pouring and to reduce the falling distance of the molten metal, reducing mechanical and thermal damage to the vessel refractory lining [2, 3]. When HM is poured inside the BOF vessels, emissions in the form of flames, hot metallic

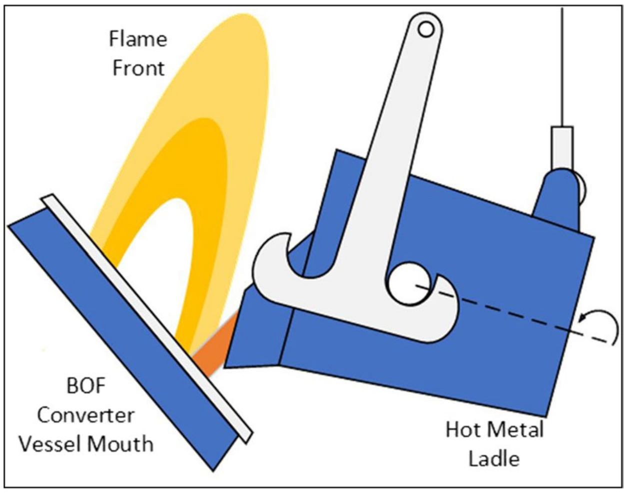

Fig. 1 A diagram representing a ladle pouring HM into a BOF vessel

fumes and dusts are released into the atmosphere due to its impact with the refractory lining and scrap metal. Excessive release of these emissions can result in adverse operational and environmental effects, causing thermal damage to the surrounding assets (e.g. crane ropes and structural components) and accumulation of metallic dusts on the floor. Multiple factors can lead to excess release of emissions, including but not limited to pouring height, rate of pouring and metallic scrap quality and size. Fume extraction systems are typically employed to mitigate damages; however, they cannot provide 100% efficiency meaning release of emissions should be reduced at the source [4]. Pouring is typically a manually operated task, where the pouring height and rate are controlled by the crane operators. Due to the human nature aspect, variability in these factors can exist. Automated monitoring and analysis of these factors lead to optimisation opportunities, standardisation of the pouring process and reduction of emissions. Automated monitoring can be achieved using computer vision techniques and uncovers the potential for a fully automated control system; however, it brings its own set of problems particularly due to the visually noisy environment that ladle pouring occurs in.

"Computer vision" is the collection of tasks that enable computers to understand visual data and has gained popularity in recent years due to its strong position in modern technology with applications related to health and medicine, self-driving cars, drones, robotics and surveillance and monitoring. Currently, computer vision capabilities are vast and rapidly growing year-on-year. Internet of Things (IoT) data collection has catalysed and popularised the development of Big Data and therefore its application with deep learning. This has led to a wide variety of constantly evolving deep learning techniques existing to accomplish tasks requiring a complex understanding of visual data. Machine learning describes the use of statistical models that can self-learn to recognise patterns in data. Deep learning is a sub-field of machine learning in which multi-layer neural networks are utilised to interpret large and/or complex data. Convolutional neural networks (CNNs) are capable of learning spatial information, making them useful for visual recognition tasks including classification, localisation, detection, segmentation and tracking. Computer vision challenges have normally been addressed using CNNs, resulting in them becoming an underpinning of most modern-day visual recognition algorithms. Region proposal networks (RPNs), which propose areas objects may be contained in, are also an effective tool for visual tasks and a key component in the well-established R-CNN algorithm family.

Ongoing research and development has led to a variety of deep learning applications in the manufacturing industry such as 3D position measurement, screw hole recognition, surface defect detection, bearing fault diagnosis and automated laser cladding bead geometry estimation [5–8]. Traditional artificial neural networks (ANNs) have even been used to intelligently reschedule the steelmaking-continuous-casting production process [9]. Furthermore, applications of computer vision have been developed across different areas of manufacturing such as mechanical machining quality inspection, automotive wheel alignment, self-calibration of 3D printing processes, fibre defect detection, electronics defect detection, tile alignment and anti-counterfeiting technologies [5]. Despite this, the application of computer vision for real-time tracking and condition-based monitoring of industrial processes is inhibited by the quality and quantity of data. For instance, poor lighting conditions of the surrounding environment and presence of dust and fumes, combined with the lack of labelled data, can result in poor prediction capabilities. Acquiring sufficiently sized labelled datasets is difficult in any industry due to the laborious nature of data-labelling; however, some of the harsh manufacturing environments such as the type steelmaking occurs in amplify this struggle.

This paper addresses the gap in remote condition monitoring of industrial processes using computer vision, by proposing a novel methodology that combines contrast enhancement, segmentation and Kalman filtering to track ladle movement and estimate furnace flame severity during ladle pouring. The dataset used to train and test the segmentation network is original and produced by labelling real-world data from steelworks. Process input parameters (pouring height and rotation angle) and a process quality indicator (flame severity) are estimated, providing a basis for analysing the relationship between process inputs and process quality. This work provides a baseline for using deep learning to advance remote condition monitoring technology specifically for ladle pouring with minimal data requirements. This paper contributes in the following ways:

- A literature review discussing techniques available for contrast enhancement, object segmentation and object tracking.
- An algorithm capable of tracking ladles in dark environments with a segmentation mAP of 0.61.
- A tool capable of estimating ladle pouring process parameters (pouring height and rotation angle) and resulting process quality (flame severity), which is highly valuable for root-cause analysis, process optimisation, predictive maintenance and closed-loop control.

This paper is organised as follows: Section 2 is a literature review that presents and discusses the main techniques used as well as potential alternatives. Section 3 describes the methodology, Section 4 presents and discusses results and Section 5 concludes the key outcomes.

2 Literature review

The first challenge of this research was to attenuate the poor lighting conditions in the video data captured at the steelworks by using a contrast enhancement technique. With clearer data, a suitable off-the-shelf segmentation network needed to be identified for application. Finally, a tracking method was required to improve segmentation performance across sequential frames.

This literature review will outline existing contrast enhancement, object segmentation and object tracking methods that could contribute to tracking moving machinery in noisy environments. Contrast enhancement is used for denoising to improve image clarity, object segmentation is used to isolate the object and object tracking promotes the success of this across sequential video frames. Instance segmentation networks which can differentiate between instances of the same class will be targeted rather than semantic segmentation networks which cannot, due to their advantage when scaling up and diversifying across different manufacturing technologies.

2.1 Denoising methods

Histogram equalisation (HE) is a commonly used method for image processing and can be used for contrast enhancement [10]. HE works by equalising the histogram of the intensity range of an image by spreading out the most frequent intensity values [11]. Overall, this increases the contrast globally and therefore causes areas of lower local contrast to gain a higher contrast [11]. HE can be performed on luminance and value channels (pixel properties in hue-saturation-value and hue-saturation-luminance colour space) as well as brightness, without affecting image hue or saturation [11].

Basic HE has been used for contrast enhancement of images of thick composites to aid the detection and characterisation of manufacturing defects [12], as well as for depth images of human bodies to obtain more information for human motion detection for product line optimisation in a computer assembly factory [13].

Adaptive histogram equalisation (AHE) takes local spatial information into consideration by dividing the image into tiles and equalising the histogram for each tile [11]. AHE has been used with deep learning models to improve the performance of glass product surface defects during the quality control process [14].

Contrast-limited adaptive histogram equalisation (CLAHE) clips the contrast of each equalised tile to prevent the over amplification of noise [11]. Excess from clipping is then redistributed over each histogram bin which prevents over-enhancement and edge-shadowing, two flaws that sometimes result from using AHE [11].

CLAHE has been used to improve image contrast for quality assessment of flat 3D-printed surfaces where it was considered successful [15]. It has also contributed to an algorithm used for automatic defect recognition in welded joints where it was considered very effective for finely sized defects [16], as well as being used successfully for enhancing microstructure images of friction stir welded joints [17]. In a previously mentioned paper related to surface inspection of plastic injection moulding, CLAHE was used to improve results and whilst it improved precision, it was detrimental to recall and AP [18]. However, the paper reports that in practice, using CLAHE solved malfunction problems that previously existed within the system [18].

Brightness Preserving Bi-Histogram Equalisation (BPBHE) preserves brightness better than standard HE by using two sub-images that capture pixels with the lower and upper halves of the intensity range respectively, equalises and then combines them [19]. BPBHE has been used for fault detection of a hot metal body where it helped intensify the appearance of hot spots making them easier to recognise [20].

2.2 Segmentation methods

Mask R-CNN is a segmentation network released in 2017 that belongs to the R-CNN family of deep learning networks. Its predecessor Faster R-CNN is an object detection network that when trained, can predict bounding boxes that encapsulate detected objects. It does this by first proposing "regions of interest" which are likely to contain an object and then classifying each proposal as containing an object or the background [21]. Mask R-CNN mainly builds on this by possessing an additional capability of predicting a mask that supersedes the bounding box prediction by predicting with pixel-level accuracy, as opposed to a simple box prediction [22].

The Microsoft Common Objects in Context (MSCOCO) challenge has existed since 2014 and is based on the COCO dataset [23]. The performance evaluation metric is average precision (AP), which is also named mAP depending on the source. These two metrics are often interchanged and presented as percentages or decimals, but ultimately represent two different metrics. This is clarified mathematically in Sect. 3. For the COCO segmentation challenge, Mask R-CNN has achieved an AP of 0.37 which is superior to the winning 0.29 AP achieved by FCIS (Fully Convolutional Instance-aware Semantic Segmentation) in 2016 and the winning 0.25 AP by MNC (Multi-Task Network Cascade) in 2015 [22].

Mask R-CNN is an exceptional choice for industry application, not only for the performance it provides, but also for the fact it is well documented and therefore more straightforward to develop and optimise compared to its peers. Applications of Mask R-CNN in the manufacturing industry are vast and include automated machine inspection when combined with augmented reality, where accuracies of 70% and 100% were achieved depending on which machines were inspected [24], automated defect detection during powder spreading in selective laser melting with 92.7% accuracy and 0.22 s per image [25], solder joint recognition with over 0.95 mAP [26], identification and tracking of objects in manufacturing plants in near-real time leading to automatic object misplacement identification where a precision (number of correct predictions compared to number of overall predictions) of 0.99 and a recall (number of correct predictions compared to number of ground truth instances) of 0.98 were achieved [27], classification and localisation of semiconductor wafer map defect patterns with 97.7% accuracy [28], detection and segmentation of aircraft cable brackets with an AP of 0.998, recall of 99.5% and mean intersection-over-union (mIoU, an average of the IoU metric explained in Sect. 3) of 84.5% with a time of 1.02 s per bracket compared to the traditional method which took ten seconds [29], surface defect detection of automotive engine parts with an mAP of 0.85 component assembly inspection with a classification accuracy of 86.6% [30], welding deviation detection in keyhole TIG deep penetration welding with a satisfactory outcome of ± 0.133 mm and variance of 0.0056mm^2 [31], automated pointer meter reading with an AP of 0.71, automated defect detection of industrial filter cloth with an accuracy of 87.3% [32] and finally, wind turbine blade defect detection and classification with an AP, using a 0.5 IoU threshold, of 82.6% [33].

Based on competitiveness in research challenges and successful application in many areas of manufacturing industry, it is sensible to state that Mask R-CNN is a strong choice for anyone considering implementation of computer vision into manufacturing sites. Whilst it could be argued that there is currently no instance segmentation network as well documented with the same proven ability in industrial applications, there are several other models that have exhibited competency.

YOLACT (”you only look at coefficients”) is a one-stage instance segmentation network as opposed to the two stages of Mask R-CNN (region proposal and classification). Two tasks are performed in parallel: generation of “prototype masks” over the entire image (similar to region proposals) and prediction of coefficients per mask [34]. Instance masks are then created by linearly combining prototype masks and mask coefficients [34].

On the COCO dataset, several variations of YOLACT were evaluated against each other and other models. The most competitive variation achieved an AP of 0.298 which was slightly higher than the recorded 0.295 AP of FCIS and lower than the recorded 0.36 AP of Mask R-CNN [34]. However, YOLACT ran at 33.5fps (frames per second) as opposed to Mask R-CNN that ran at 8.6fps [34].

YOLACT has been applied within the manufacturing industry to some extent, with notable applications being surface inspection of plastic injection moulded tampon applicators which achieved a precision of 0.49, a recall of 0.56 and an AP of 0.4 at an IoU threshold of 0.5 [18], metal screw defect segmentation with a COCO mAP of 0.41 [35], detection of metal screw head defects with an accuracy of 92.8% and a detection speed of 0.03 s per image [36], automated automotive part assessment where an mAP of 0.67 was achieved [37] and finally, more generally but still applicable to the manufacturing industry, is safety zone estimation and violation detection for nonstationary objects in workplaces, where YOLACT achieved a segmentation accuracy of 97.7% on one dataset and 96.8% on another [38]. Furthermore, YOLACT + +, the successor of YOLACT, was modified and used for surface defect segmentation of magnetic tiles where it achieved an AP of 0.27 and speed of 12.4fps, in comparison to the original YOLACT + + AP of 0.25 and speed of 11.8fps, the original YOLACT AP of 0.23 and speed of 11.5fps and the SOLO (“segmenting objects by locations”) AP of 0.23 and speed of 10.6fps [39].

SOLO, another one-stage instance segmentation network, was evaluated on the COCO dataset and achieved an mAP of 0.38 which was reported as the same as Mask R-CNN, whilst superior to every other model evaluated which includes but is not limited to YOLACT, FCIS and MNC [40]. Its successor SOLOv2 has been used for automatically identifying geometrical parameters of self-piercing riveting joints with 0.98 mean IoU over various samples [41].

PANet (Path Aggregation Network) is a successor of Mask R-CNN won the MSCOCO challenge in 2017 and has shown improved performance over Mask R-CNN on the COCO dataset (0.42 segmentation mAP compared to 0.37 for Mask R-CNN), as well as outperformed various other models on other datasets [42]. PANet is less prevalent than

Mask R-CNN with regard to implementation and industrial application; however, it has been used for multi-sided surface defect detection of polyethylene (PE), polypropylene (PP) and acrylonitrile butadiene styrene (ABS) and achieved 0.98 mAP and 0.98 recall [43].

BlendMask is an instance segmentation network that combines instance-level information with semantic information and has surpassed Mask R-CNN on the COCO dataset with an mAP of 0.38 whilst Mask R-CNN was again reported at 0.37, and whilst doing so, BlendMask performed approximately 20% faster with an inference time of 0.074 s whilst Mask R-CNN was reported at over 0.097 s [44]. A variation of BlendMask has been used successfully for fastener counting in automated production where it achieved 5.02% error in estimating the quantity of fasteners present in a dense dataset and 5.69% error in a sparse dataset [45].

Ariadne + is an instance segmentation network that combines deep learning with graph theory and was used to segment instances of wires with a mean IoU of 0.79, an AP of 0.66 and a speed of 0.36 s [46]. The same task was achieved by another model called FASTDLO (Fast Deformable Linear Objects) which claims to surpass Ariadne + in both speed and IoU (no mention of AP), however it achieved a lower IoU of 0.78 and a significantly better speed of 0.046 s [47].

2.3 Tracking methods

Long short-term memory networks (LSTMs) are neural networks that can retain information over one or more timesteps and are therefore very useful when dealing with sequential data such as videos (since they are essentially a sequence of frames) [48]. The memory they possess makes them useful for tracking-based tasks and in the case of video data, they are often combined with CNNs to capture spatial and temporal information simultaneously.

A bidirectional LSTM network has been used for trajectory tracking and prediction of thrown objects as part of a smart manufacturing system that uses throwing and catching robots to accelerate the transportation of manufacturing parts with speeds of up to 10 ms^{-1} over distances of up to 3 m, with a maximum error achieved of no more than 2 mm [49].

LSTMs have also been used to track additive manufacturing processes to prevent cyber-physical attacks that compromise mechanical properties and functionalities with a precision of 0.95, recall of 0.98 and computation time of 0.85 ms [50]. Furthermore, they have been used for tracking and prediction of the remaining useful life of manufacturing machines with a root mean square error of 15.42 cycles which was the lowest of six models [51], and also tracking the trajectory of piezoelectric actuators which reduced the maximum tracking error of the closed-loop system from 1.59 to 0.15 μm (90.4% reduction) [52].

Like LSTMS, CNN-LSTMs have been used for equipment health condition recognition and prediction and have achieved 98.6% test accuracy [53]. They have also been used for monitoring weld penetration from dynamic weld pool serial images with 0.3 mm accuracy [54] and power data pattern detection and tracking for manufacturing sites with a test loss of 0.1197, where loss is a measure of how different predictions are to the real values [55].

A Kalman filter is an algorithm that uses measured values over a series of timesteps in conjunction with an initial “guess”, to predict the future state of a system. Whilst the first guess is less informed and likely to be incorrect, the algorithm follows a two-step process of firstly making a prediction and then secondly retrieving the measurement update and correcting itself based on the error between the two. This process is repeated as a system is active and results in accurate tracking of a given variable (which could be position).

Kalman filters have been used in a range of manufacturing applications, both with and without the involvement of machine learning. Kalman filters were combined with a fuzzy expert system and incorporated into a fastening tool tip tracking system, which allowed identification of fastened bolts [56]. Here, Kalman filters were specifically used for tool orientation estimation and tool center of mass location estimation [56]. The model was successful and reduced tool position error significantly when eight bolts were fastened during an experiment [56]. The operator-only approach and the developed algorithm resulted in final position errors of 93 mm and 6 mm respectively [56]. Robust detection of weld position and seam tracking on weld pool images was also achieved with Kalman filtering, where the weld position covariance error was reduced from 0.0084 to 0.0010 mm, whilst the seam tracking error was reduced from 0.33 to 0.11 mm [57]. Kalman filtering was also used for tool flank wear estimation when cutting gamma-prime strengthened alloys, where it reduced root mean square error by 41% on one experiment, increased error by 8% in a repeat experiment and then reduced error by 25% in a third repetition [58]. An angle and position tracking system for semi-automated manufacturing processes was achieved using a Kalman filter–based approach and resulted in an overall tracking accuracy of 3.18 cm [59].

To mention a few variants of the traditional Kalman filter, an extended Kalman filter was used for tool flank wear area estimation in wet-turning of Inconel 718 and increased the accuracy of estimation by a maximum of 60% [60]; a multi-rate Kalman filter was used for damage detection in composite beams by tracking the neutral axis under different loading conditions where it was successful across a range of static loads, dynamic loads and temperatures [61]. The required threshold to avoid false negative predictions was surpassed with the direct estimate method’s standard deviation; however, with the Kalman filter method, the standard deviation was much smaller and within the threshold [61]. Additionally, an adaptive Kalman filter was used to sense contact force and torque for robotic manipulators for manufacturing tasks, where the root mean square error for force estimation ranged from approximately 0.78 to 1.35N and for torque estimation it ranged from approximately 0.12 to 0.18Nm [62].

Kalman filters have also been used jointly with machine learning for a range of manufacturing applications. A network based on AlexNet (a well-known image classification network) was used in parallel with Kalman filtering to detect chatter in milling and achieved a 98.9% accuracy [63]; a CNN was combined with Kalman filtering to track steel sheet coils during transport to the uncoiler at eight frames per second with a deviation below 15 pixels [64] and tool wear condition monitoring during turning was achieved using an artificial neural network together with an extended Kalman filter which resulted in a classification accuracy of 89.2% [65]. Furthermore, a deformation force monitoring method for aero-engine casing machining was developed using a combination of a deep autoregressive network and Kalman filtering, which improved the success rate of monitoring by approximately 30% compared to the traditional approach and the deformation calculation based on the predicted deformation force was less than 0.008 mm [66].

3 Methodology

The task was to, with minimal resources, segment a hot metal ladle frame-by-frame with a mAP of at least 0.5 and therefore track activity during pouring, whilst also estimating pour quality. The methodology followed is summarised in Fig. 2. The flow chart shows six steps. Firstly, data preparation consisted of frame labelling and frame denoising.

Initial training and testing of Mask R-CNN (pre-trained on the COCO dataset) followed which used a NVIDIA GeForce RTX 2070 Super graphics processing unit (GPU).

Next, Stage 1 and 2 optimisations followed which each comprised of experiments that tested a range of settings for most hyperparameters of Mask R-CNN which are listed in Fig. 2. Stage 1 initially tuned some hyperparameters individually and then tuned two groups of hyperparameters using an optimisation method. The method was inspired by Taguchi's orthogonal arrays which are used in Design of Experiments (DoE) methodology to plan the study of many variables simultaneously whilst conducting the minimal number of experiments [67], as well as the grid search optimisation method which consists of choosing a set of values for each variable and then exhaustively conducting experiments that test every possible combination of values [68]. The idea behind this method was that the grid search has a disadvantage of being extremely inefficient whilst orthogonal arrays are very efficient and therefore eliminate the weakness of the grid search. For the purpose of this paper, this optimisation method will be referred to as "efficient grid search" (EGS). Hyperparameter groups were tested in sequence whilst carrying the best model along each group. The EGS was used for the entirety of Stage 2 and the process of grouping hyperparameters and cascading the model is novel.

Kalman filter tracking began by firstly sampling points along the predicted mask edge and using them as input to multiple Kalman filters that tracked the predicted mask shape. The state transition matrix, a key element to the Kalman filter performance, was initially chosen arbitrarily and improved through pure trial-and-error. A convex hull was also produced using the gift-wrapping algorithm to create a smooth mask. The way Kalman tracking was implemented to improve the mask prediction of Mask R-CNN is also novel.

Development of the industrial application was the final step which comprised of firstly setting a fixed offset from the mask to the actual ladle boundary as this was observed to be consistent, before setting a 4-point moving average to smooth fluctuations across frames. Secondly, the bottom edge of the ladle was predicted most consistently so this was captured using a sliding window, used to derive a line of best fit and then the angle of the new line was used to calculate the rotation angle estimate. This was smoothed using a 5-point moving average. Finally, the flame severity was estimated by counting the number of pixels in the flame region with a brightness value above 235.

3.1 Metrics

3.1.1 Intersection-over-union, precision and recall

During non-maximum suppression (NMS), windows overlaying an image were scored on their likelihood of containing the ladle. The highest scoring window was kept and windows overlapping it by more than a given intersection-over-union (IoU) threshold were suppressed. Of the remaining windows not kept or suppressed (not overlapping the highest scoring window, or overlapping by less than the threshold), the highest scoring window was also kept and windows overlapping by the threshold value or more were suppressed. This process was repeated until all windows were kept or suppressed [69]. For each remaining prediction, the IoU with ground truth masks was calculated as in (1) [70]:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{1}$$

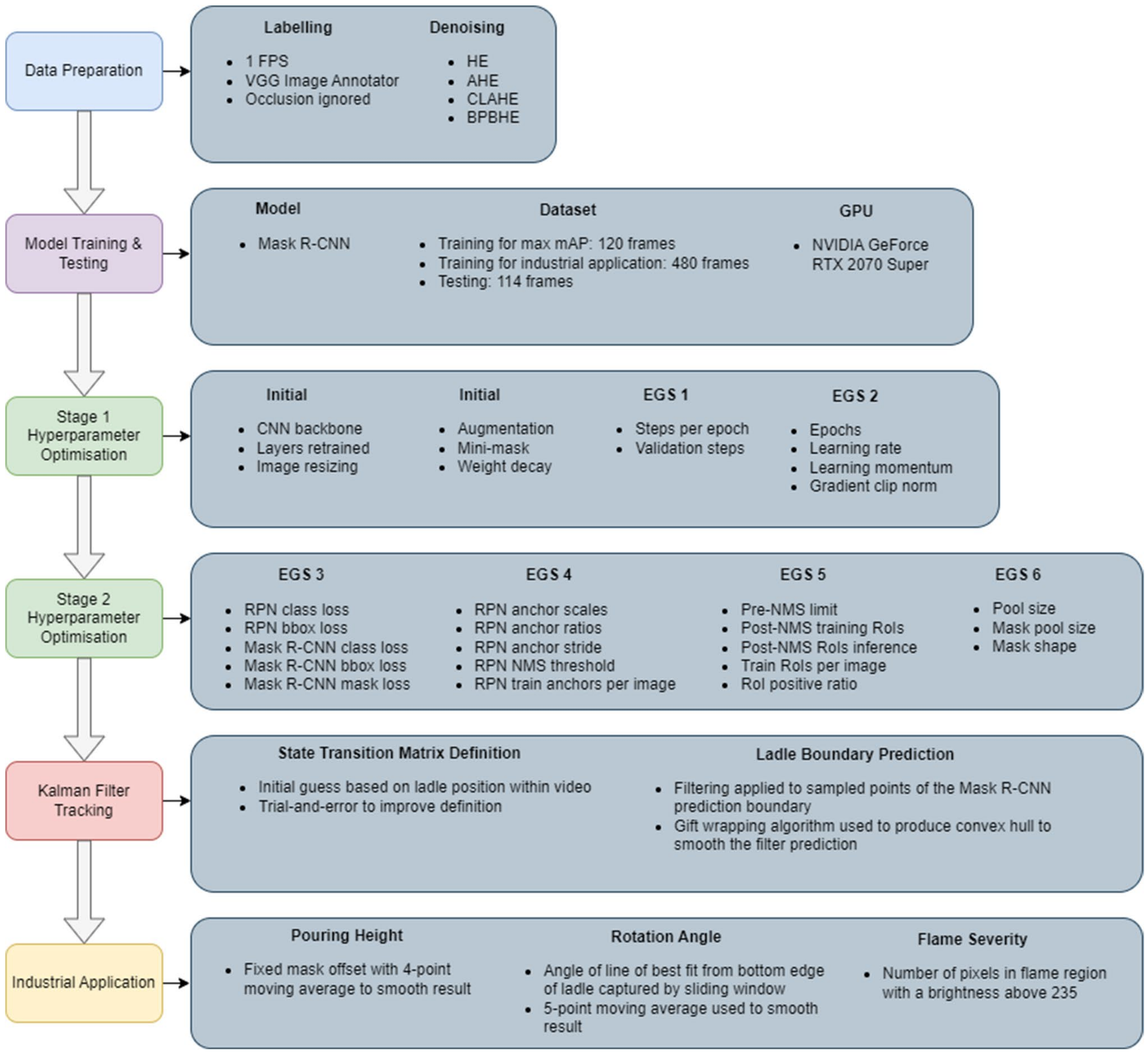
Data Preparation
Labelling
1 FPS
VGG Image Annotator
Occlusion ignored
Denoising
HE
AHE
CLAHE
BPBHE
Model Training & Testing
Model
Mask R-CNN
Dataset
Training for max mAP: 120 frames
Training for industrial application: 480 frames
Testing: 114 frames
GPU
NVIDIA GeForce RTX 2070 Super
Stage 1 Hyperparameter Optimisation
Initial
CNN backbone
Layers retrained
Image resizing
Initial
Augmentation
Mini-mask
Weight decay
EGS 1
Steps per epoch
Validation steps
EGS 2
Epochs
Learning rate
Learning momentum
Gradient clip norm
Stage 2 Hyperparameter Optimisation
EGS 3
RPN class loss
RPN bbox loss
Mask R-CNN class loss
Mask R-CNN bbox loss
Mask R-CNN mask loss
EGS 4
RPN anchor scales
RPN anchor ratios
RPN anchor stride
RPN NMS threshold
RPN train anchors per image
EGS 5
Pre-NMS limit
Post-NMS training RoIs
Post-NMS RoIs inference
Train RoIs per image
RoI positive ratio
EGS 6
Pool size
Mask pool size
Mask shape
Kalman Filter Tracking
State Transition Matrix Definition
Initial guess based on ladle position within video
Trial-and-error to improve definition
Ladle Boundary Prediction
Filtering applied to sampled points of the Mask R-CNN prediction boundary
Gift wrapping algorithm used to produce convex hull to smooth the filter prediction
Industrial Application
Pouring Height
Fixed mask offset with 4-point moving average to smooth result
Rotation Angle
Angle of line of best fit from bottom edge of ladle captured by sliding window
5-point moving average used to smooth result
Flame Severity
Number of pixels in flame region with a brightness above 235

Fig. 2 A flow chart showing the methodology used

This was used to evaluate the number of true positives (TP) and false positives (FP) as follows in (2) and (3) respectively:

$$TP \text{ if } IoU > threshold \tag{2}$$

$$FP \text{ if } IoU > threshold \tag{3}$$

Precision and recall could be calculated using the total number of true positive, false positive and false negative (FN) results as shown in (4) and (5), where FN is the number of ground truths that were not predicted [71]:

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{\#predictions} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{\#ground\ truths} \tag{5}$$

3.1.2 Average precision

Precision and recall values for all TP predictions on all test images were ordered based on increasing recall and used to

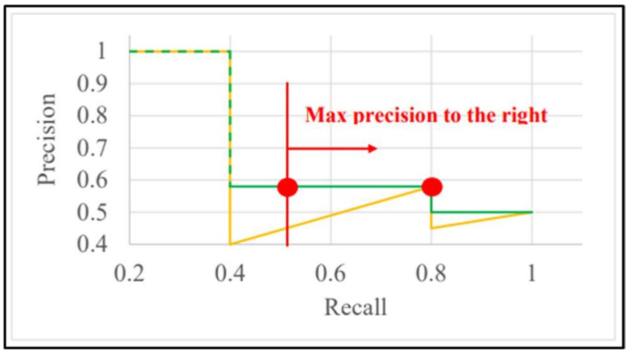

Fig. 3 Example of smoothing a precision-recall curve [72]

plot a precision-recall curve like the orange one shown in Fig. 3 [69].

Before calculating the AP, the curve was smoothed by replacing the precision at every data point with the maximum precision value with the same or higher recall value, resulting in the green line in Fig. 3 [69].

This is a conventional approach based on the fact that each relevant prediction shows increased precision with increased recall (up and right). Also, irrelevant predictions show no change, or a decrease in precision. Predictions with non-maximum precision values for the recall value are considered irrelevant because they do not change the AP [69].

For the COCO standard evaluation, AP was calculated by dividing the curve up at 101 points along the recall axis, summing the 101 corresponding precision values and then taking the mean average. This process is shown mathematically in (6) [73].

$$AP = \frac{1}{101} \sum_{r \in (0.0,\ldots,1.0)} AP_r = \frac{1}{101} \sum_{r \in (0.0,\ldots,1.0)} p_{interp}(r) \tag{6}$$

P_r represents the precision at each selected recall point and $p_{interp}(r)$ is the precision at each selected recall point after smoothing, shown in (7) where $\widetilde{r}$ is the selected recall point.

$$p_{interp}(r) = \underset{\widetilde{r}:\widetilde{r} \geq r}{max} p(\widetilde{r}) \tag{7}$$

If this was a multi-class problem, the previous step would be done for every class and an average of AP values would be taken which would then be named "mean average precision" (mAP). However, since this problem uses one class (the ladle), this value was named "average precision" instead, which was to reserve the name "mean average precision" for the COCO-based mAP discussed below.

3.1.3 Mean average precision

COCO standard evaluation was used to reduce bias and was conducted by using a range of IoU threshold values to calculate a range of corresponding AP values. Threshold values range from 0.5 to 0.95 in increments of 0.05. Upon calculation of all AP values, the mean average of them was taken and named the mean average precision. This is what was considered mAP for this paper. The formula is shown in (8) [72].

$$mAP_{coco} = \frac{mAP_{0.50} + mAP_{0.55} + \ldots mAP_{0.95}}{10} \tag{8}$$

The mAP calculated in (8) was calculated initially for each individual frame, so could be renamed as in (9).

$$mAP_{frame} = \frac{mAP_{0.50} + mAP_{0.55} + \ldots mAP_{0.95}}{10} \tag{9}$$

The final mAP performance achieved on one video was the mean average of the mAP values for all frames, as shown in (10), where n was the number of frames.

$$mAP_{sequence} = \frac{\sum mAP_{frame}}{n} \tag{10}$$

3.2 Data preparation

3.2.1 Frame labelling

Before labelling, frames were extracted from videos using a frame extraction code at one frame per second, which was a minimalist approach when considering the raw video ran at over 30fps and this was done to reduce the computational requirements and complexity of the task, whilst still dealing with a visually sufficient output for any observers. Labelling of frames was achieved using Visual Geometry Group Image Annotator (VIA) [74]. This process consisted of drawing a polygon (mask) line-by-line that encapsulated the ladle sensibly, then copy-and-pasting this polygon across to the next frame, modifying it accordingly and then repeating the process until every frame was labelled.

There were objects occluding the ladle at certain points of the video, so it was necessary to make extra considerations for this. To deal with occlusions there were three options:

- The ladle could be labelled as if occlusions were non-existent.
- Only the main visible part of the ladle could be labelled.
- All visible parts of the ladle could be labelled as either complete or partial.

The first option was suitable. The second option was attempted but was unsuitable as Mask R-CNN would ideally predict as close to a fixed shape mask as possible, and occlusions would dramatically change the mask shape with this approach. The third option was extremely time

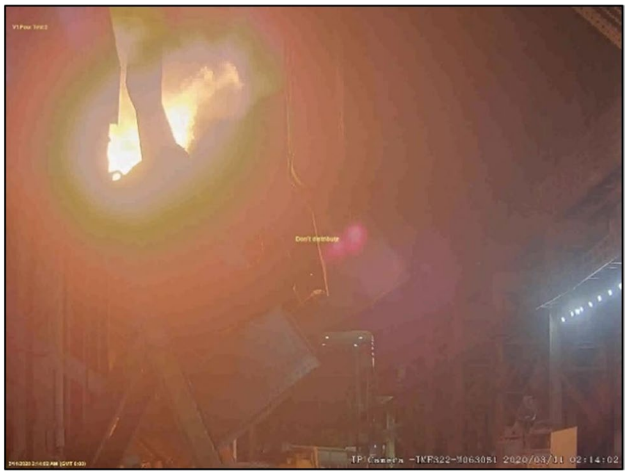

Fig. 4 A frame with poor lighting, noticeable mostly around the right-side edge of the ladle

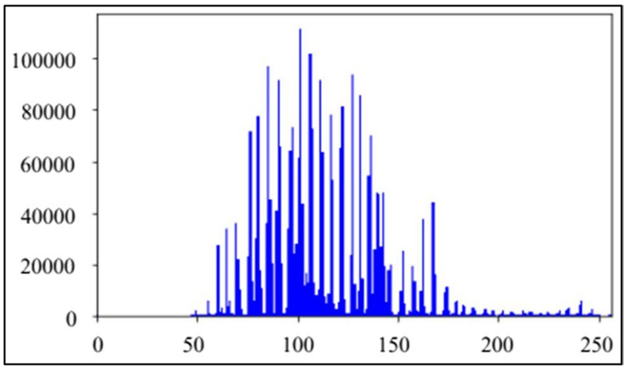

Fig. 5 A greyscale histogram of the image in Fig. 4 that shows the 256 potential greyscale values along the x-axis and pixel frequency of each intensity value on the y-axis

consuming and would promote mask predictions varying largely in shape across frames.

3.2.2 Frame denoising

As shown in Fig. 4, during the video, the lighting was quite poor and therefore there were frames where the ladle was distorted. Since this made Mask R-CNN performance worse, attempts were made to reduce the distortion by improving the lighting. Contrast enhancement was approached by first converting frames from RGB (red-green-blue) to greyscale and then using variants of histogram equalisation to improve the lighting whilst maintaining the frame quality. The variants of histogram equalisation used are described below.

Standard Histogram Equalisation To perform histogram equalisation, it was first necessary to convert frames to greyscale and obtain the intensity values for each pixel, which ranged from 0 to 255 and represented how black (0) or white (255) a pixel is.

Afterwards, a histogram was created based on the pixel frequency of each intensity. Figure 5 shows the histogram of Fig. 4.

Once a histogram such as the one in Fig. 5 was produced, it was possible to enhance the contrast by equalisation which is essentially using a wider range of intensity values more frequently.

Firstly, the probability mass function (PMF), which is the probability of each intensity value occurring within the image, was calculated. This was done by dividing the frequency of each intensity value from the histogram by the total number of pixels within the frame, as described in (11). In (11), $p(x_i)$ is the probability of any given greyscale value X occurring, s is the frequency of occurrence the given greyscale value and S is the total number of pixels in the frame [75].

$$p(x_i) = P(\{s \in S | X(s) = x_i\}) \tag{11}$$

From the PMF, the cumulative distribution function (CDF) could be calculated, which is the cumulative sum of all PMF values and is shown in (12), where $F(x)$ represents the CDF [75].

$$F(x) = P(X \leq x), for\ any\ x \in \mathbb{R} \tag{12}$$

Each CDF value (calculated accumulatively along each PMF value) was then multiplied by the corresponding number of grey levels minus one to obtain new intensity values [75].

Adaptive histogram equalisation AHE is a development of the standard HE method and works by first dividing the image up into evenly sized squares and then applying standard HE to each square separately. This creates a block-like effect with regard to the contrast of the image which is countered using bilinear interpolation [76].

Contrast-limited adaptive histogram equalisation Since AHE is prone to emphasising noise when it stands out from a square containing pixels with mostly similar intensity values, contrast limiting can be applied prior to calculation of the CDF [77]. Contrast limiting is setting a frequency threshold and clipping the histogram at that point prior to CDF to avoid over-enhancement of noise present in homogenous regions. Contrast limiting is appropriate mainly for images with a relatively even distribution of intensity values [78].

Brightness preserving bi-histogram equalisation HE can change the mean brightness of an image which may result in poor quality. As the name states, BPBHE was developed to preserve the brightness of an equalised image. BPBHE performs HE on two sub-images produced by decomposition of the original image using the mean intensity value [10]. One sub-image includes the pixels with intensity less than or equal to the mean value; the other contains pixels with intensity greater than the mean. The lower intensity sub-image is equalised with the constraint that new pixel intensity values are mapped into the range of minimum intensity level to mean, and the higher intensity sub-image is equalised with the constraint that new pixel intensity values are mapped into the range of mean intensity value to maximum. Finally, the two sub-images are combined to produce the result [10].

3.3 Mask R-CNN

Mask R-CNN was customised for ladle tracking in two stages. Firstly, hyperparameters were optimised for maximum mAP. For those with discrete input, all available settings were tested, whereas those with a continuous range of potential values were tested using the EGS. The order in which hyperparameters were tested was crucial in maximising the effectiveness of the optimisation stages. In the second stage, a selection of hyperparameters were grouped into cascading EGS setups which each optimised mAP before the optimal model for each EGS was carried forward to the next setup.

3.3.1 Stage 1 optimisation

The hyperparameters optimised in Stage 1 can be seen in Table 1 in order of experimentation. The CNN backbones readily available were ResNet-50 and ResNet-101. The backbone was pre-trained on the COCO dataset meaning it was likely to be inefficient to retrain the whole backbone on new data. Therefore, the re-training options were to retrain the layers constituting the network heads, Stages 3 and up, Stages 4 and up, Stages 5 and up or all layers.

The network heads were used for classification and regression of bounding boxes, and mask prediction [22]. Stages 3–5 refer to five stages of ResNet-101 and four stages of ResNet-50, which are groups of the 101 and 50 respective layers of the ResNet backbone [79]. The backbone and retrained layers were tested first as they could significantly affect performance and training time. Also, the backbone is used first in Mask R-CNN.

Following the backbone and retrained layers in order of impact was image resizing and then augmentation testing. Mini-mask settings were tested next as it was thought they could be significantly limiting mask capabilities by reducing the overall mask size to smaller than what is required to segment the ladle. Weight decay, or L2 regularisation, was tuned next because it had more potential impact on performance than many of the other untested parameters. Mini-mask and weight decay settings that lie within the range of reasonable values were selected for experimentation. The default Mask R-CNN mini-mask size is (56, 56). Early testing showed this could have been too small, so the default as well as two larger sizes were chosen. For weight decay, values are typically between 0 and 0.1 and it is common to test this at varying orders of magnitude [80].

The remainder of both optimisation stages consisted of experimental setups using the EGS, which uses a grid search style approach but with orthogonal arrays that minimise the number of experiments required, to optimise a given number of parameters with a given number of settings [81]. For example, the first setup for steps per epoch and validation steps had two parameters and three settings and the orthogonal array chosen is called L9(3^2) and had nine experiments.

Table 1 Hyperparameters and settings for Stage 1 optimisation

Hyperparameter	Settings
CNN backbone	ResNet-50, ResNet-101
Layers retrained	Heads, 3 +, 4 +, 5 +, all
Image resizing	None, square1024, square2048, pad1024, pad2048
Augmentation	Off, on
Mini-mask	Off, on-56, on-112, on-224
Weight decay	0.01, 0.001, 0.0001
EGS 1	
Steps per epoch	250, 500, 1000
Validation steps	10, 25, 50
EGS 2	
Epochs	10, 20, 30
Learning rate	0.0001, 0.001, 0.02
Learning momentum	0.5, 0.9, 0.99
Gradient clip norm	2.5, 5, 10

3.3.2 Stage 2 optimisation

For Stage 2 optimisation, 18 hyperparameters were grouped holistically and settings were chosen similarly to before. The groups include loss weights, RPN-based parameters, RoI-based parameters and mask and pooling–based parameters. Similarly to backbone with Stage 1, all four EGS setups in Stage 2 were ordered based on when they are used in Mask R-CNN. Despite the mini-mask being tested relatively early in Stage 1, the normal mask was tested in the last EGS of Stage 2 as mask size did not appear to be limiting performance. The best model from each experimental stage was taken forward to the next stage. The hyperparameters and settings for the Stage 2 optimisation can be seen in Table 2. For loss weights it was logical to set weights to low integers to, for example, third, halve or double the weight of the losses.

Table 2 Hyperparameters and settings for Stage 2 optimisation

Hyperparameter	Settings
EGS 3	
RPN class loss	1, 2, 3
RPN bbox loss	1, 2, 3
Mask R-CNN class loss	1, 2, 3
Mask R-CNN bbox loss	1, 2, 3
Mask R-CNN mask loss	1, 2, 3
EGS 4	
RPN anchor scales	(16, 32, 64, 128, 256) (32, 64, 128, 256, 512) (64, 128, 256, 512, 1024)
RPN anchor ratios	(0.25, 0.5, 1) (0.5, 1, 2) (1, 2, 4)
RPN anchor stride	1, 2, 3
RPN NMS threshold	0.7, 0.8, 0.9
RPN train anchors per image	128, 256, 512
EGS 5	
Pre-NMS limit	3000, 6000, 12,000
Post-NMS training RoIs	1000, 2000, 4000
Post-NMS RoIs inference	500, 1000, 2000
Train RoIs per image	100, 200, 400
RoI positive ratio	0.25, 0.33, 0.5
EGS 6	
Pool size	7, 14, 28
Mask pool size	14, 28, 56
Mask shape	28, 56, 112

For EGS 4 and EGS 5, most default settings of Mask R-CNN for hyperparameters such as RPN anchor scales, anchor ratios and RPN train anchors per image were halved and doubled to define three settings for each hyperparameter. Anchor stride, NMS threshold and RoI positive ratio are exceptions to this as stride is commonly tested at values of one, two and three, NMS threshold is usually tested at values between 0.6 and 1.0 and RoI positive ratio was 0.33 at default so 0.25 and 0.5 were tested arbitrarily as they were not considered unreasonable.

For the final EGS, following the larger mask testing previously discussed with regard to the mini-mask, the normal mask settings were doubled twice too.

3.4 Kalman filter tracking

After Mask R-CNN optimisation, there was an attempt to further evolve the tracking ability of the model by incorporating Kalman filter edge refinement. A Kalman filter is a mathematical tool used to predict the future movement of an object. It consists of five Eqs. (13–17) that work recursively to predict the unknown nature of a system, which is the solution of the least-squares method [82].

The filter works by first guessing/predicting the state of the system then updating that guess based on feedback control from noisy measured data. The prediction step uses (13) and (14) to obtain the current state and error covariance estimates to obtain what is called the a priori estimate for the next time step. During the update step (15–17), a new measurement is incorporated into the a priori estimate to get an a posteriori estimate [82]. This is explained in more detail below.

In (13), the first element of the predict step, the a priori estimate for the state of the next step $\widehat{x}_{k+1}^{-}$ is calculated by adding the product of the state transition matrix A_k and the a priori estimate for the current step $\widehat{x}_k$, to the product of the control input matrix and the control vector Bu_k [82].

$$\widehat{x}_{k+1}^{-} = A_k\widehat{x}_k + Bu_k \tag{13}$$

In (14), the a priori estimate of the error covariance of the next step $\widehat{P}_{k+1}^{-}$ is calculated by the adding the product of the transition matrix and the a posteriori estimate of the error covariance of the current step P_k, to the transition matrix and process noise Q_k [82].

$$\widehat{P}_{k+1}^{-} = A_kP_k + A_k^T + Q_k \tag{14}$$

In (15), which is the first element of the update step, the optimal Kalman gain K_k is calculated using the a priori estimate of the error covariance of the current step, P_k^-, the Jacobian matrix H_k and the measurement error covariance R_k [82].

$$K_k = P_k^-H_k^T\left(H_kP_k^-H_k^T + R_k\right)^{-1} \tag{15}$$

In (16), the a posteriori state estimate $\widehat{x}_k$ is calculated using the a priori state estimate $\widehat{x}_k^-$, the Kalman gain K and the actual measurement z_k [82].

$$\widehat{x}_k = \widehat{x}_k^- + K(z_k - H_k\widehat{x}_k^-) \tag{16}$$

In (17), the a posteriori estimate of error covariance P_k is calculated using an identity matrix I, the Kalman gain, the Jacobian matrix and the a priori error covariance for the current step [82].

$$P_k = \left(I - K_kH_k\right)P_k^- \tag{17}$$

The idea of this was to use one Kalman filter for each point within a collection of points sampled from the edges of the originally predicted mask. It was thought that each filter would use the corresponding edge point pixel co-ordinates from each successive frame as measured input. Then, following the update step the Kalman prediction would be an improvement on the original Mask R-CNN prediction when all new point predictions are used collectively, which would improve overall mAP.

Edge pixels of the original Mask R-CNN mask were detected using image processing and their co-ordinates were saved in a list. Varying numbers of points were sampled and used as initial input for the Kalman filters. Each Kalman filter tracked a single point which resulted in an arrangement of Kalman point predictions that could change based on the Kalman filter setup.

Using the gift-wrapping algorithm, a convex hull was created using the outer points. The convex hull ensured a smoother mask edge was captured around the outside of the point predictions, rather than a jagged one that would capture less of the ladle. The convex hull (Fig. 6) was the tightest, or most efficient boundary that encapsulated all points whilst having all interior angles below 180° [84].

Following the convex hull, the state transition matrix of the Kalman filters were improved through experimentation. The state transition matrix relates the system state at the current time step to the state in the next time step, essentially indicating how to move from one to the other [82].

The state transition matrix in this scenario is 4×4 in size and describes the motion of the points from frame to frame. Since this problem was 2D, there was an x-position, x-velocity, y-position and y-velocity, each represented by one row of the transition matrix. The elements in each row corresponded to terms in the 2D kinematic equations, terms for which the values can change based on what is known [85].

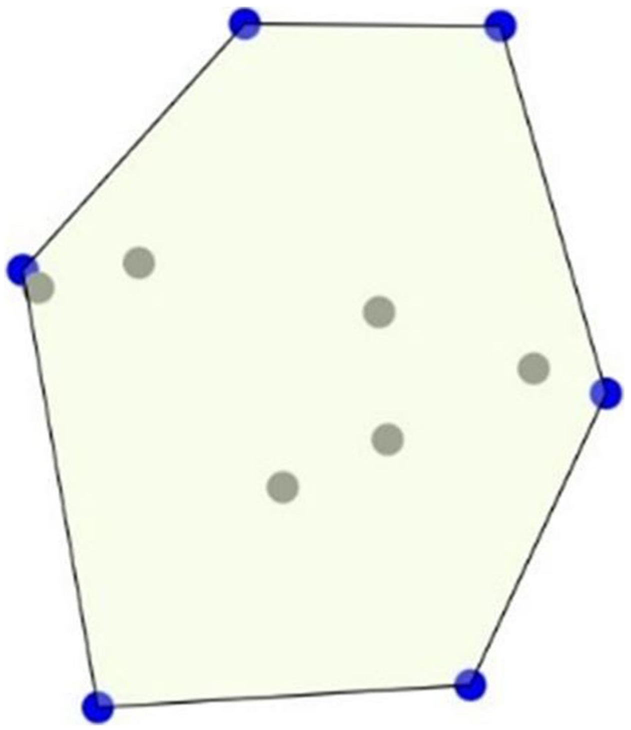

Fig. 6 An example of a convex hull [83]

Since no exact transition values were known, experimentation began by increasing the default number for x_{k-1} from one to ten in increments of one and then keeping the value which resulted in the highest mAP. This process was repeated for y_{k-1} whilst keeping the best value found for x_{k-1}, then this process was repeated for the final two variables.

The default state transition matrix is denoted A_k in (13). Each element equal to one is a multiplier for the corresponding terms in $\hat{x}_k$, which are the x-position, y-position, x-velocity and y-velocity in descending order of (18):

$$\hat{x}_k = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} \tag{18}$$

Therefore, changes to elements [1, 1, 2, 2, 3, 3] and [4, 4] in the transition matrix affected how the Kalman filter made predictions.

3.5 Industrial application

To adapt the model for industry application, three variables to describe the ladle movement and the flames were estimated: pouring height, rotation angle and flame severity. Pouring height and rotation angle were measured as they could be used to calculate tilting force and were strong factors affecting pour quality. Flame severity is an indicator of pour quality because less severe flames result in less equipment degradation, reduced danger and reduced emissions. By estimating process inputs and pour quality simultaneously and automatically, this work provides an efficient way to observe and therefore improve pour quality, meaning necessary costs for repairing damage or purchasing new equipment will be less frequent, as well as reduced injuries, reduced emissions and a consistent pouring technique with minimal human input.

For pouring height, the top pixel of the predicted mask with a vertical correction to compensate for the difference between mask edge and ladle edge was used.

For rotation angle, a moving window captured the angle of the bottom edge of the ladle. The window moved relatively to the angle of the previous frame so that the bottom ladle edge was continually captured as it rotated. Five and 4-point moving averages were used for pouring height and rotation angle estimation respectively, in order to make the measurement more robust to fluctuations in mask edge positions.

For flame severity, the denoised frames applied were cropped to only contain the area significantly affected by the furnace flames. Pixels of the remaining image area with a brightness of over 235 were counted and the total amounts for each frame were used to give a severity rating.

4 Results and discussion

4.1 Denoising data

Initial experiments showed an improvement of mAP with denoising, compared to non-denoised experiments. In Fig. 7, the left column contains "good contrast" images, the middle column contains "moderate contrast" images and the right column "poor contrast". Row A is after converting to greyscale but before denoising, and the remaining rows each show the results of the various denoising techniques described in Section 3.

The aim of denoising was to ensure more distinct ladle visibility so the model could learn and predict better. Important details to consider when evaluating what the results in Fig. 7 show are clarity of ladle edges and the brightness of the furnace flame. Results from HE (row (b)) show a significant improvement with regard to ladle edge visibility compared to the greyscale images. This is especially evident in the poor contrast image, where the top-right quarter of the ladle changes from almost completely invisible to clearly lighter than the background. However, with regard to the furnace flame brightness, lens flare is significantly increased. Looking at the poor image, despite the overall change in image aesthetic, increased lens flare does not appear to be distorting any important aspects of the image. Contrarily, the good image could be considered worse than before HE application, as the ladle edges, particularly in the top-right quarter, are less distinct than before.

Standard HE smooths out the histogram of the intensity values for each pixel effectively reducing extremities, and the bright furnace flame is a large extremity compared to the surrounding dark pixels.

Therefore, since HE equalises over the whole image at once, the brightness of the furnace flame is distributed over the rest of the image. This is effective for the poor image because it is extremely dark everywhere except the flame location, so this is a good "source of light". However, in the poor image, there is already some clarity present in the image before HE. Dispersing the extreme brightness from the flame to the already relatively bright pixels causes some extreme brightness to occur farther away from the flame, which includes pixels constituting the ladle edges.

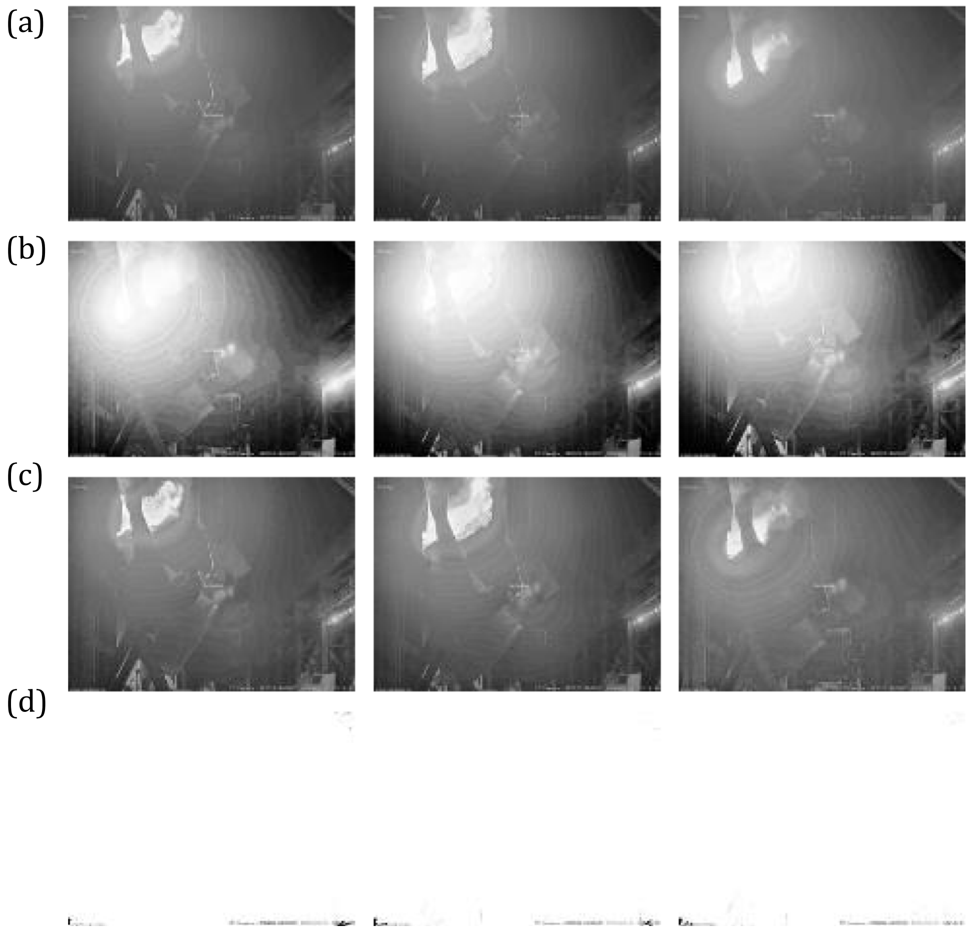

Fig. 7 Results of the various denoising techniques, with examples for good (left), moderate (center) and poor (right) quality contrast frames. **a** Original greyscale, **b** HE, **c** CLAHE and **d** BPBHE

CLAHE (row (c)) equalises multiple histograms representing divided “tiles” of the image to prevent over-enhancement of noise in homogeneous regions, which is the issue with the HE results.

Therefore, a significantly improved and more balanced performance across varying contrast qualities is observable in the CLAHE results. Figure 8 shows clearly the difference in edge visibility before and after CLAHE.

BPBHE (row (d)) is designed to preserve the mean brightness of an image, and in this case, application has resulted in bright white images for all three contrast qualities. This indicates that the mean intensity value is high (higher intensity pixels are whiter), so the higher intensity sub-image would contain extremely bright pixels and upon equalisation, produce a white image. The lower intensity sub-image would contain some moderately dark pixels and some very bright pixels, resulting in a bright image. When the sub-images are combined, intensity values are very high, which explains the extreme brightness that can be seen.

The best technique appeared to be contrast-limited adaptive histogram equalisation (CLAHE) and so the performance of Mask R-CNN when trained and tested on data denoised using this method was compared to model performance when using the original data. The results of this comparison are shown in Table 3.

4.2 Mask R-CNN Optimisation

Table 4 presents optimal settings for Stage 1 hyperparameters. ResNet-101 was superior to ResNet-50 due to extra layers capturing more features during feature extraction, so that Mask R-CNN has a better understanding of the data. ResNet-101 runs at 7.6×10^9 floating operations per second (FLOPs), whereas ResNet-50 only runs at 3.8×10^9 FLOPs [86].

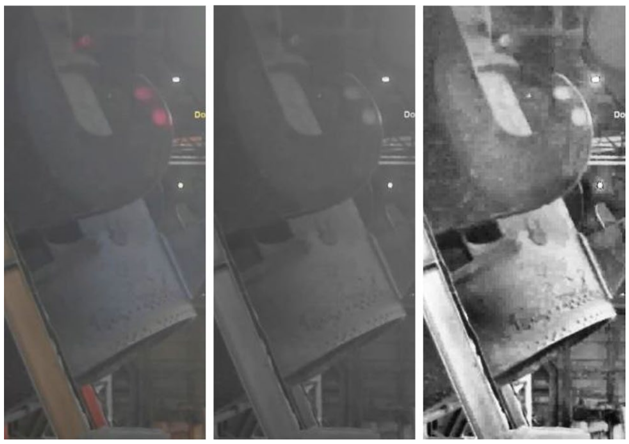

Fig. 8 Ladle edge in original video (left), greyscale video (center) and denoised video (right)

Table 3 mAP comparison pre-optimisation for original and denoised

	Original	Denoised
mAP	0.0044	0.1115

Re-training stages 3 and up of ResNet-101 (layer 12 to layer 101) was likely optimal because it allowed the backbone to retain some understanding of commonly recognisable features learned from being pre-trained on the COCO dataset, whilst leaving the majority of the relatively large network for learning information specific to the ladle data.

Using the pad64 image resizing option with a size of (1024, 1024) was better than the square setting because the ladle shape is irregular, asymmetric and a distinguishable for feature extraction; therefore, shape makes it less recognisable. Whilst producing a square image, pad64 preserves original shapes by completing the square shape with empty padding. Augmentation also relies on the model learning from distorted data.

Mini-mask application was not optimal. The mini-mask efficiently captures the mask within the bounding box only, whereas the normal mask captures the mask within the whole image [87]. Whilst this sped training up by roughly 12%, it reduced the mAP. The fact that the ladle takes up a relatively large proportion of the image, compared to say, a person in an image of a room full of people, means that the mini-mask had an insignificant effect on performance. Default settings of weight decay, steps per epoch, validation steps, learning rate, momentum and gradient clip norm were already optimal. Not every hyperparameter needs tweaking between different applications and so it seems that change of these hyperparameters is unnecessary. The mAP achieved

Table 4 Hyperparameters with corresponding optimal settings for Stage 1 optimisation

Hyperparameter	Settings
CNN backbone	ResNet-101
Layers retrained	3+
Image resizing	Pad1024
Augmentation	Off
Mini-mask	Off
Weight decay	0.0001
EGS 1	
Steps per epoch	100
Validation steps	50
EGS 2	
Epochs	Variable
Learning rate	0.001
Learning momentum	0.9
Gradient clip norm	5

after Stage 1 was 0.28. This model was trained for 20 epochs which took just under 23 min each on average and just under 7.6 h in total. The inference time for each frame was 0.41 s.

Table 5 contains the results of Stage 2 hyperparameter optimisation. The optimal loss weight combination was when losses were mostly equal; however, the Mask R-CNN classifier head and RPN bounding box losses required less weighting compared to other losses. Whilst the RPN class loss classifies the foreground and background of anchor boxes, the Mask R-CNN class loss classifies the actual objects. The only classes present were the ladle and the background, so classification was not a particularly challenging aspect of the task for the classifier head, whereas initially determining the ladle from the background is more challenging. This explains why RPN class loss remains equal to Mask R-CNN bounding box and mask losses whilst Mask R-CNN class loss is reduced to one. With regard to the RPN bounding box loss, this is for locating objects within an image, whereas the Mask R-CNN bounding box loss is for locating classified objects within an image. Therefore, since there is only one object, the RPN has essentially done the job, making Mask R-CNN bounding box loss less important. This means the RPN bounding box loss should be at three and the Mask R-CNN bounding box loss should be the same or reduced. The combinations (3, 3, 1, 3, 3), (3, 3, 1, 2, 3) or (3, 3, 1, 1, 3) would be optimal based on this discussion; however, they were not part of the EGS due to the predefined orthogonal setup.

The number of train anchors and anchor scales appeared optimal at default, RPN anchor ratios were halved and RPN anchor strides were increased from one to two. The NMS threshold increased from 0.7 to 0.9. Anchors essentially locate objects, so the aim was to optimise their size, movement and initial quantity to locate and learn objects optimially. Since the ladle is quite a large object relative to image size, initial expectations would be that ratios and stride would likely increase. Stride has increased, but ratios have decreased. This could be due to the unique ladle shape creating a need to capture the smaller details comprising the edges, which could also be why the number of train anchors and anchor scales have not increased. The increase to NMS threshold is significant and indicates requiring 20% more of an overlap of boxes to suppress them, so of the boxes present, more were needed for feature extraction which was again possibly due to the ladle shape.

The pre-NMS limit and post-NMS training RoIs remained at default; however, post-NMS inference RoIs doubled, train RoIs per image halved and the RoI positive ratio decreased from 0.33 to 0.25. The same number of RoIs was kept prior to NMS; however afterwards, the model used twice as many when making an inference, which could be explained by ladle edges requiring extra RoIs to capture them. The number of RoIs per image passed to the classifier and mask heads during training was halved and the percentage of positive RoIs used for training decreased from 33 to 25%. This indicates that, whilst the model needed to learn more detail about each frame, the object classification head and mask head required less RoIs for training. This could be explained by the increase in NMS threshold in the previous EGS, which would output a larger pool of RoIs meaning the top 6000 would be more likely to have higher objectness (likelihood of containing an object), which means less RoIs per image and a lower RoI ratio could result in a similar amount overall due to more positive RoIs existing per image. Furthermore, better quality RoIs could reduce the need for as high of a quantity.

Pool size remained at default and mask pool size quadrupled which makes sense as the ladle mask is complex so probably requires more RoIs. Mask shape also quadrupled for the same reason. The mAP was 0.52 after Stage 2. This model was trained for ten epochs which took just under 4 min each on average and just under 38 min in total. The inference time for each frame was 0.46 s.

Table 5 Hyperparameters with corresponding optimal settings for Stage 2 optimisation

Hyperparameter	Settings
EGS 3	
RPN class loss	3
RPN bbox loss	2
Mask R-CNN class loss	1
Mask R-CNN bbox loss	3
Mask R-CNN mask loss	3
EGS 4	
RPN anchor scales	(32, 64, 128, 256, 512)
RPN anchor ratios	(0.25, 0.5, 1)
RPN anchor stride	2
RPN NMS threshold	0.9
RPN train anchors per image	256
EGS 5	
Pre-NMS limit	6000
Post-NMS training RoIs	2000
Post-NMS RoIs inference	2000
Train RoIs per image	100
RoI positive ratio	0.25
EGS 6	
Pool size	7
Mask pool size	56
Mask shape	112

4.3 Kalman filter

The default state transition matrix A_k is in (19), where Δt indicates the change in timestep.

Table 6 State transition matrix experiments

Variable experiment	Final mAP
x_{k-1}	0.58
y_{k-1}	0.61
$\dot{x}_{k-1}$	0.61
$\dot{y}_{k-1}$	0.61

$$A_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{19}$$

Elements [1, 1, 2, 2, 3, 3] and [4, 4] were tested in turn, whilst maintaining the optimal value for the previous element(s). The optimal matrix is shown in (20).

$$A_k = \begin{bmatrix} 6 & 0 & \Delta t & 0 \\ 0 & 8 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{20}$$

Table 6 shows how optimisation of each element affected mAP. Since the original mAP was 0.52, optimising the first element gave an improvement of 0.06, optimising the second gave an improvement of 0.03 and the final two elements did not improve performance so were left at default. The addition of the Kalman filtering increased the average inference time from 0.46 to 0.59 s.

In Table 6, x_{k-1}([1,1] in (20)) and y_{k-1} ([2,2] in (20)) are position values and $\dot{x}_{k-1}$ ([3,3] in (20)) and $\dot{y}_{k-1}$ ([4,4] in (20)) are velocity values. Position values improved the overall mAP but velocity values did not, which is likely because in terms of position, the ladle is mostly on the lower lefthand side of the field of view, and since python reads the image from top to bottom (y-coordinate increases from image top to image bottom), the six and eight in (20) roughly align with the x and y positioning respectively, whereas the velocity of each point is inconsistent and difficult to estimate.

With reference to Section 2.2, Mask R-CNN was chosen due to its reported experimental performance, but also due to the abundance of documentation and proven application to manufacturing industry. Direct comparison of the final model to state-of-the-art models was not possible due to several unique aspects of the task (denoising, the irregular ladle shape and the rotating of the ladle). However, the final model achieved a mAP of 0.61 and a speed of 1.7fps. In comparison, Mask R-CNN was predominantly reported as achieving a mAP of 0.37 on the COCO dataset and this surpassed the 2015 and 2016 challenge winners. Furthermore, on the COCO test-dev dataset, state-of-the-art models such as YOLACT [34], SparseInst [88], SOLOv2 [89], Blendmask [44] and SipMask [90] have reportedly achieved mAPs of 0.30, 0.38, 0.37, 0.41 and 0.33 respectively, of which none surpasses the mAP achieved by the model presented in this paper. A caveat is that these models do surpass the model presented in this paper in terms of inference speed; however, this is cushioned by both the unique challenges in this task combined with the massive difference in available training data (328,000 images compared to the 234 used here).

4.4 Industrial application

A sample frame of the final output of the model created can be seen in Fig. 9. For this element of the research, the training set was increased from 120 to 480 images to maximise performance, and the Kalman filtering was disabled as it was found to reduce the consistency of the ladle shape across frames, which was detrimental to the pouring height and rotation angle estimations. In future work, a fixed ladle shape could be positioned using the centroid of the predicted mask and angled based on the estimated rotation angle.

In Fig. 9, the pouring height measurement is indicated by the dark blue horizontal line. The pixel unit can be converted to real distance units relative to the environment shown in the image using distance estimation methods. The rotation angle measurement is observable in the top-left corner of the image where the light blue line is what is captured from the bottom edge of the ladle by the moving window, and the yellow line is the line of best fit. Flame severity was calculated but not visualised as it was unnecessary and slowed the model down.

Every value seen in Fig. 9 is recorded for every frame and added to a.csv file containing the results for the entire footage. Steelmaking industry can benefit from this model in two main ways. Firstly, pouring height and rotation angle can be used to calculate tilting force, a process input parameter that affects the flame severity.

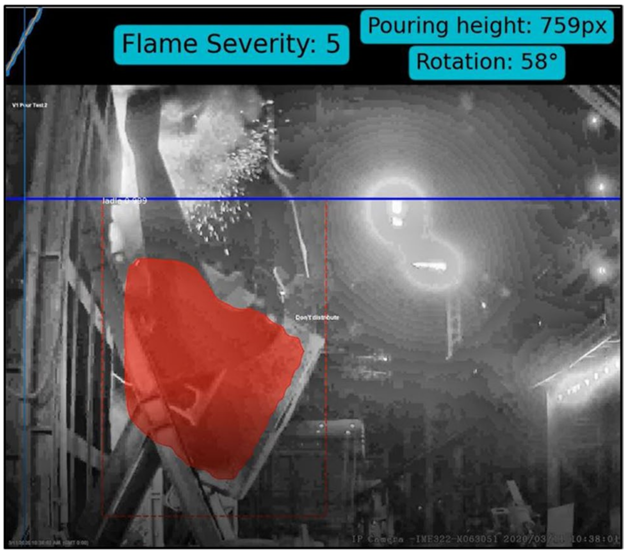

Fig. 9 A frame with measurements displayed

Using this data, the input tilting force and resulting flame severity can be easily compared. The individual trends of tilting force and flame severity combined with the relationship between them can be used to discover process quality improvement methods. Secondly, this model takes the first steps into revolutionising the integration of computer vision into steelmaking. This model paves the way for subsequent evolution of advanced technology by laying a foundation for ladle-related process optimisation that can be built upon through addition of capabilities such as closed-loop control systems that minimise emission severity and damage from accident occurrence, data collection capabilities for predictive maintenance and root-cause analysis by measuring pouring frequency and accumulative flame damage per ladle, and insightful capabilities into process variables such as ladle velocity, acceleration, process duration and energy expenditure which could all be used for process optimisation.

5 Conclusions

In this paper, noisy ladle pouring video data was denoised and used to train a segmentation model capable of tracking the shape of the ladle. Many model hyperparameters were optimised through extensive experimentation and Kalman tracking was added to further improve performance. The model was also adapted for industrial application by using the predicted mask and other image information to estimate real process parameters.

When denoising data, four different histogram equalisation methods were tested and CLAHE performed best overall when considering frames of good, moderate and poor contrast quality.

From Stage 1 optimisation of Mask R-CNN, re-training ResNet-101 from Stages 3 and up, with a weight decay of 0.0001 and a learning rate of 0.001, was found to be the optimal setup, which achieved a mAP of 0.28 and an inference time of 0.41 s.

From Stage 2 optimisation, a weight loss configuration of 3, 2, 1, 3, and 3 for RPN class, RPN bounding box, Mask R-CNN class, Mask R-CNN bounding box and Mask R-CNN mask losses respectively, an RPN NMS threshold of 0.9 and a mask shape of (112, 112), resulted in the optimal setup which achieved a mAP of 0.52 and an inference time of 0.46 s.

Further improvement was achieved through Kalman filter application, which was optimal when x and y positions in the state transition matrix were 6 and 8 respectively, which when applied resulted in a mAP of 0.61 and an inference time of 0.59 s.

This research uniquely contributes to multiple fields as it brings together the established tasks of video segmentation from computer vision, and process optimisation from manufacturing, to provide a starting point for developing remote condition monitoring technologies for ladle pouring processes. Automated collection of pouring height, rotation angle and flame severity values throughout pours will provide insight into how process input (tilting force) affects process quality (flame severity). In comparison to existing studies in this area, this paper is advantageous due to providing a novel solution that has not been addressed previously. Also, it discusses the tuning of a vast range of hyperparameters which has led to an array of suggestions useful for any researchers working with Mask R-CNN in industry. In contrast, with a larger dataset model performance could be improved significantly. Furthermore, future work would incorporate a distance estimation method to bring the model closer to production standard. Overall, this work provides an automated solution to minimising emissions, equipment damage and hazard severity, whilst paving the way to measurement of a potentially wide variety of process parameters, as well as pour quality factors, to collect data beneficial for tasks such as process optimisation, predictive maintenance and closed-loop monitoring.

Funding Cinzia Giannetti and Grazia Todeschini acknowledge the support of the UK Engineering and Physical Sciences Research Council (EPSRC) projects EP/S001387/1, EP/V061798/1 and EP/T013206/2. The authors would also like to acknowledge the M2A funding from the European Social Fund via the Welsh Government (c80816), the Engineering and Physical Sciences Research Council (Grant Ref: EP/L015099/1). We further acknowledge the support of the Supercomputing Wales (c80898 and c80900) and Ser Cymru AccelerateAI projects, which are partly funded by the European Regional Development Fund (ERDF) via the Welsh Government.

Data availability This research would not have been possible without the ladle video data from Tata Steel Europe Ltd.

Code availability Not applicable.

Declarations

Ethical approval The authors state that the research conducted for this paper was conducted ethically. To the best of the authors' knowledge, there are no known conflicts of interest.

Consent to participate Not applicable.

Consent to publish Not applicable.

Conflict of interest The authors declare no competing interests.

References

1. Annarella LA (2004) Portraits of industry. The culture of work in the industrial paintings of Howard L. Worner and Their Use in Arts Education. Maryland WA: University Press of America
2. Turkdogan ET (1996) Fundamentals of steelmaking. United Kingdom: Institute of Materials
3. Freuhan RJ (1998) The making, shaping and treating of steel. Pittsburgh PA: The AISE Steel Foundation
4. Madhavan N, Brooks G, Rhamdhani MA, Bordignon A (2022) Contribution of CO2 emissions from basic oxygen steelmaking process. Metals. 12(5):1–18. Available from: https://doi.org/10.3390/met12050797
5. Zhou L, Zhang L, Konz N (2022) Computer vision techniques in manufacturing. IEEE Trans. Syst. Man Cybern. 53(1):105–17. Available from: https://doi.org/10.1109/TSMC.2022.3166397
6. Amarouayache IIE, Saadi MN, Guersi N, Boutasseta N (2020) Bearing fault diagnostics using EEMD processing and convolutional neural network methods. Int J Adv Manuf Technol. 107:4077–95. Available from: https://doi.org/10.1007/s00170-020-05315-9
7. Liang M, Cao P, Tang J (2021) Rolling bearing fault diagnosis based on feature fusion with parallel convolutional neural network. Int J Adv Manuf Technol. 112:819–31. Available from: https://doi.org/10.1007/s00170-020-06401-8
8. Gonçalves DA, Stemmer MR, Pereira M. (2019) A convolutional neural network approach on bead geometry estimation for a laser cladding system. Int J Adv Manuf Technol. 106:1811–21. Available from: 10.1007/ s00170–019–04669-z
9. Liu W, Pang X, Li H, Sun L (2022) Ladle intelligent re-scheduling method in steelmaking–refining–continuous casting production process based on BP neural network working condition estimation. Int J Adv Manuf Technol. 122:65–85. Available from: https://doi.org/10.1007/s00170-021-08327-1
10. Kaur M, Kaur J, Kaur J (2011) Survey of contrast enhancement techniques based on histogram equalization. Int J Adv Comput Sci Appl 2(7):137–41. https://doi.org/10.14569/IJACSA.2011.020721
11. Sudhakar S (2017) Histogram equalization [Internet]. [cited 2023 Jan 15]. Available from: https://towardsdatascience.com/histogram-equalization-5d1013626e64
12. Daniel M, Wooh SC (1989) Ultrasonic techniques for characterization of manufacturing defects in thick composites. Rev Prog Quant Nondestr Eval 1605–12. Available from: https://doi.org/10.1007/978-1-4613-0817-1_202
13. Lipovits Á, Gál M, Kiss JP, Süveges C (2016) Human motion detection in manufacturing process. MVML'16. 2016 Aug 16–17; Budapest. Ottawa: International ASET Inc
14. Sari F, Ulas AB (2022) Deep learning application in detecting glass defects with color space conversion and adaptive histogram equalization. Trait. du Signal. 39(2):731–6. Available from: https://doi.org/10.18280/ts.390238
15. Fastowicz J, Okarma K (2019) Quality assessment of photographed 3D printed flat surfaces using Hough transform and histogram equalization. JUCS. 25(6):701–17. Available from: https://doi.org/10.3217/jucs-025-06-0701
16. Chakraborty S, Acharya R, Gupta RK, Mishra VK, et al. (2017) Computer aided defect detection strategy for welded joints. eJNDT. 104–9. Available from: https://www.ndt.net/article/nde-india2016/papers/A145.pdf
17. Mishra A. (2021) Contrast limited adaptive histogram equalization (CLAHE) approach for enhancement of the microstructures of friction stir welded joints [Internet]. [cited 2023 Jan 15]. Available from: https://arxiv.org/ftp/arxiv/papers/2109/2109.00886.pdf
18. Im D, Lee S, Lee H, Yoon B, et al. (2021) A data-centric approach to design and analysis of a surface-inspection system based on deep learning in the plastic injection molding industry. Processes 2021. 9(11):1–22. Available from: https://doi.org/10.3390/pr9111895
19. Kim YT (1997) Contrast enhancement using brightness preserving bi-histogram equalization. IEEE Trans Consum Electron 43(1):1–8. https://doi.org/10.1109/30.580378
20. Maji D, Biswas M, Nath S, Bhaduri S. (2013) Fault detection of a hot metal body. IJSRD. 1(8):1535–6. Available from: https://www.academia.edu/6826611/Fault_Detection_of_A_Hot_Metal_Body
21. Khan S, Rahmani H, Shah SAA, Bennamoun M (2018) A guide to convolutional neural networks for computer vision. Synth lect comput vis 8:1–207. https://doi.org/10.2200/S00822ED1V01Y201712COV015
22. He K, Gkioxari G, Dollár P, Girshick R. (2017) Mask R-CNN. ICCV 2017. 2017 Oct 22–29; Venice. New York: IEEE
23. Microsoft. MSCOCO challenge leaderboard [Internet]. 2023 [cited 2023 Jan 15]. Available from: https://cocodataset.org/#detection-leaderboard
24. Perdpunya T, Nuchitprasitchai S, Boonrawd P (2021) Augmented reality with Mask R-CNN (ARR-CNN) inspection for Intelligent Manufacturing. IAIT2021. 2021 Jun 29 – Jul 1; Bangkok. New York: ACM
25. Chen HY, Lin CC, Horng MH, Chang LK, et al. (2022) Deep learning applied to defect detection in powder spreading process of magnetic material additive manufacturing. Materials 2022. 15(16):1–12. Available from: https://doi.org/10.3390/ma15165662
26. Wu H, Gao W, Xu X (2019) Solder joint recognition using Mask R-CNN method. IEEE Trans Compon Packaging Manuf Technol. 10(3):525–30. Available from: https://doi.org/10.1109/TCPMT.2019.2952393
27. Karim MM, Doell D, Lingard R, Yin Z, et al. (2019) A region-based deep learning algorithm for detecting and tracking objects in manufacturing plants. Procedia Manufacturing. 39:168–77. Available from: https://doi.org/10.1016/j.promfg.2020.01.289
28. Chiu MC, Chen TM. (2021) Applying data augmentation and Mask R-CNN-based instance segmentation method for mixed-type wafer maps defect patterns classification. IEEE Trans Semicond Manuf 34(4):455–63. Available from: https://doi.org/10.1109/TSM.2021.3118922
29. Zhao G, Hu J, Xiao W, Zou J (2021) A mask R-CNN based method for inspecting cable brackets in aircraft. CJA 34(12):214–26. Available from: https://doi.org/10.1016/j.cja.2020.09.024
30. Zhu H, Wang Y, Fan J (2022) IA-Mask R-CNN: improved anchor design Mask R-CNN for surface defect detection of automotive engine parts. Appl Sci 2022. 12(13):1–15. Available from: https://doi.org/10.3390/app12136633
31. Chen Y, Shi Y, Cui Y, Chen X (2021) Narrow gap deviation detection in Keyhole TIG welding using image processing method based on Mask-RCNN model. Int J Adv Manuf Technol 112:2015–25. Available from: https://doi.org/10.1007/s00170-020-06466-5
32. Zuo L, Peilin H, Changhua Z, Zhehan Z. (2020) A robust approach to reading recognition of pointer meters based on improved mask R-CNN. Neurocomputing. 388:90–101. Available from: https://doi.org/10.1016/j.neucom.2020.01.032
33. Zhang J, Cosma G, Watkins J. (2021) Image enhanced Mask R-CNN: a deep learning pipeline with new evaluation measures for wind turbine blade defect detection and classification. J Imaging 2021. 7(3):1–20. Available from: https://doi.org/10.3390/jimaging7030046
34. Bolya D, Zhou C, Xiao F, Lee YJ. (2020) YOLACT: real-time instance segmentation. ICCV 2019. 2019 Oct 27 – Nov 2; Seoul. New York: IEEE

35. Chen WY, Tsao YR, Lai JY, Hung CJ, et al. (2022) Real-time instance segmentation of metal screw defects based on deep learning approach. Measurement Science Review. 22(3):107–11. Available from: https://doi.org/10.2478/msr-2022-0014
36. Lai JY, Tsao YR, Liu CY. (2022) High-accuracy detection and classification of defect and deformation of metal screw head achieved by convolutional neural networks. Appl Mech Mater 909:75-80. Available from: https://doi.org/10.4028/p-fy36nq
37. Yusuf SA, Aldawsari AA, Souissi R (2022) Automotive parts assessment: applying real-time instance-segmentation models to identify vehicle parts [Internet]. [cited 2023 Jan 15]. Available from: https://doi.org/10.48550/arXiv.2202.00884
38. Cho H, Lee K, Choi N, Kim S, et al. (2022) Online safety zone estimation and violation detection for nonstationary objects in workplaces. IEEE Access. 10:39769–39781. Available from: https://doi.org/10.1109/ACCESS.2022.3165821
39. Wen H, Chen L, Fu T, Yang Z, et al. (2022) Detecting the surface defects of the magnetic-tile based on improved YOLACT ++. ICCT 2021. 2021 Oct 13–16; Tianjin. New York: IEEE
40. Wang X, Kong T, Shen C, Jiang Y, et al. (2020) SOLO: segmenting objects by locations. ECCV 2020. 2020 Aug 23-28; Online. Berlin: Springer
41. Li M, Liu Z, Huang L, Chen Q, et al. (2022) Automatic identification framework of the geometric parameters on self-piercing riveting cross-section using deep learning. J. Manuf. Process. 83:427–37. Available from: https://doi.org/10.1016/j.jmapro.2022.09.020
42. Liu S, Qi L, Qin H, Shi J, et al. (2018) Path aggregation network for instance segmentation. CVPR 2018. 2018 Jun 18–23; Salt Lake City. New York: IEEE
43. Han Y, Jeong J (2020) Real-time inspection of multi-sided surface defects based on PANet model. ICCSA 2020. 2020 Jul 1-4; Cagliari. Berlin: Springer
44. Chen H, Sun K, Tian Z, Shen C, et al. (2020) BlendMask: top-down meets bottom-up for instance segmentation. CVPR 2020. 2020 Jun 13–19; Seattle. New York: IEEE
45. Qi K, Zhu Q, Zhou X, Wang Y, et al. (2022) Fastener counting method with an improved Blendmask. ASCC 2022. 2022 May 4–7; Jeju. New York: IEEE
46. Caporali A, Zanella R, Greogrio DD, Palli G (2022) Ariadne+: deep learning–based augmented framework for the instance segmentation of wires. IEEE Trans Industr Inform. 18(12):8607–17. Available from: https://doi.org/10.1109/TII.2022.3154477
47. Caporali A, Zanella R, Greogrio DD, Palli G (2022) FASTDLO: fast deformable linear objects instance segmentation. IEEE Robot Autom Lett 7(4):9075–82. Available from: https://doi.org/10.1109/LRA.2022.3189791
48. O'Donovan C, Giannetti C, Todeschini G (2021) A novel deep learning power quality disturbance classification method using autoencoders. ICAART 2021. 2021 Feb 4-6; Vienna. Vienna: SCITEPRESS
49. Qadeer N, Shah JH, Sharif M, Khan MA, et al. (2022) Intelligent tracking of mechanically thrown objects by industrial catching robot for automated in-plant logistics 4.0. Sensors 2022. 22(6):1–21. Available from: https://doi.org/10.3390/s22062113
50. Shi Z, Mamun AA, Kan C, Tian W, et al. (2022) An LSTM-autoencoder based online side channel monitoring approach for cyber-physical attack detection in additive manufacturing. J Intell Manuf Available from:https://doi.org/10.1007/s10845-021-01879-9
51. Zhang J, Wang P, Yan R, Gao RX. (2018) Long short-term memory for machine remaining life prediction. J Manuf Syst 48(C):78-86. Available from: https://doi.org/10.1016/j.jmsy.2018.05.011
52. Li J, Huang Y, Li Q, Li Y (2022) Closed-LSTM neural network based reference modification for trajectory tracking of piezoelectric actuator. Neurocomputing. 467:379–91. Available from: https://doi.org/10.1016/j.neucom.2021.10.012
53. Wang H, Fu S, Peng B, Wang N, et al. (2021) Equipment health condition recognition and prediction based on CNN-LSTM deep learning. IncoME-V 2020. 2020 Apr 15-17; Zhuhai. Berlin: Springer
54. Yu R, Kershaw J, Wang P, Zhang YM (2022) How to accurately monitor the weld penetration from dynamic weld pool serial images using CNN-LSTM deep learning model? IEEE Robot Autom Lett 7(3):6519–25. Available from: https://doi.org/10.1109/LRA.2022.3173659
55. Lee JH, Kang J, Shim W, Chung HS, et al. (2020) Pattern detection model using a deep learning algorithm for power data analysis in abnormal conditions. Electronics 2020. 9(7):1–18. Available from: https://doi.org/10.3390/electronics9071140
56. Won SHP, Golnaraghi F, Melek WW (2008) A fastening tool tracking system using an IMU and a position sensor with Kalman filters and a fuzzy expert system. IEEE Trans. Ind. Electron. 56(5):1782–92. Available from: https://doi.org/10.1109/TIE.2008.2010166
57. Gao XD, Na SJ. (2005) Detection of weld position and seam tracking based on Kalman filtering of weld pool images. J Manuf Syst 24(1):1–12. Available from: https://doi.org/10.1016/S0278-6125(06)00002-1
58. Niaki FA, Ulutan D, Mears L (2015) In-process tool flank wear estimation in machining Gamma-prime strengthened alloys using Kalman filter. Procedia Manufacturing. 1:696–707. Available from: https://doi.org/10.1016/j.promfg.2015.09.018
59. Lipka M, Meinel D, Müller S, Sippel E, et al. (2020) A wireless angle and position tracking concept for live data control of advanced, semi-automated manufacturing processes. Sensors 2020. 20(9):1–15. Available from: https://doi.org/10.3390/s20092589
60. Niaki FA, Michel M, Mears L (2016) State of health monitoring in machining: extended Kalman filter for tool wear assessment in turning of IN718 hard-to-machine alloy. J. Manuf. Process. 24(2):361–9. Available from: https://doi.org/10.1016/j.jmapro.2016.06.015
61. Soman R, Majewska K, Mieloszyk M, Malinowski P, et al. (2018) Application of Kalman Filter based Neutral Axis tracking for damage detection in composites structures. Composite Structures. 184:66–77. Available from: https://doi.org/10.1016/j.compstruct.2017.09.092
62. Cao F, Docherty PD, Ni S, Chen XQ (2021) Contact force and torque sensing for serial manipulator based on an adaptive Kalman filter with variable time period. Robot Comput Integr Manuf 72:1-13. Available from: https://doi.org/10.1016/j.rcim.2021.102210
63. Rahimi MH, Huynh HN, Altintas Y (2021) On-line chatter detection in milling with hybrid machine learning and physics-based model. CIRP Manuf. Sci. Technol. 35:25-40. Available from: https://doi.org/10.1016/j.cirpj.2021.05.006
64. Xu K, Zheng G, Zhang F. (2021) Tracking steel coils using CV algorithm based on deep learning and Kalman filter. ICCAIS 2021. 2021 Oct 14–17; Xi'an. New York: IEEE
65. Purushothaman S (2009) Tool wear monitoring using artificial neural network based on extended Kalman filter weight updation with transformed input patterns. J Intell Manuf 21(6):717–30. Available from: https://doi.org/10.1007/s10845-009-0249-y
66. Guo H, Li Y, Liu C, Ni Y, et al. (2022) A deformation force monitoring method for aero-engine casing machining based on deep autoregressive network and Kalman filter. Appl. Sci. 2022. 12(14):1–20. Available from: https://doi.org/10.3390/app12147014

67. Roy RK (2020) Design of experiments using the Taguchi approach: 16 steps to product and process improvement. John Wiley & Sons, Hoboken
68. Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. JMLR. 13:281–305. Available from: https://jmlr.csail.mit.edu/papers/volume13/bergstra12a/bergstra12a.pdf
69. Henderson P, Ferrari V (2016) End-to-end training of object class detectors for mean average precision. ACCV 2016. 2016 Nov 20-24; Taipei. Berlin: Springer
70. Gonzalez S, Arellano C, Tapia JE (2019) Deepblueberry: quantification of blueberries in the wild using instance segmentation. IEEE Access. 7:105776–105788. Available from: https://doi.org/10.1109/ACCESS.2019.2933062
71. Scikit-learn developers. Precision-recall [Internet]. 2007–2022 [cited 2023 Jan 15]. Available from: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html
72. Hui J (2023) mAP (mean average precision) for object detection [Internet]. 2018 Mar 7 [cited 2023 Jan 15]. Available from: https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173
73. Lin TY, Pepose S, Girshick R, Wu Y. (2019) cocoeval.py [Internet]. Dec 25 [cited 2023 Jan 15]. Available from: https://github.com/cocodataset/cocoapi/blob/master/PythonAPI/pycocotools/cocoeval.py
74. Dutta A, Gupta A, Zisserman A (2016) VIA image annotator [Internet]. 2016 Sep [cited 2023 Jan 15]. Available from: https://www.robots.ox.ac.uk/~vgg/software/via/
75. Kuter K(2020) Probability mass functions (PMFs) and cumulative distribution functions (CDFs) for discrete random variables [Internet]. 2020 Jan 9 [cited 2023 Jan 15]. Available from: https://stats.libretexts.org/Courses/Saint_Mary's_College_Notre_Dame/MATH_345__-_Probability_(Kuter)/3%3A_Discrete_Random_Variables/3.2%3A_Probability_Mass_Functions_(PMFs)_and_Cumulative_Distribution_Functions_(CDFs)_for_Discrete_Random_Variables
76. OpenCV. Histogram equalization [Internet]. 2022 Dec 29 [cited 2023 Jan 15]. Available from: https://docs.opencv.org/4.x/d5/daf/tutorial_py_histogram_equalization.html
77. Arora S, Agarwal M, Kumar V, Gupta D (2018) Comparative study of image enhancement techniques using histogram equalization on degraded images. IJET 7(8):468–71. Available from: https://doi.org/10.14419/ijet.v7i2.8.10487
78. Shome SK, Vadali SRK (2011) Enhancement of diabetic retinopathy imagery using contrast limited adaptive histogram equalization. IJCSIT. 2(6):2694–9. Available from: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=7e0f9ed873453a92c21ae1427137ac501608f1ed
79. Sachan A (2017) Detailed guide to understand and implement ResNets [Internet]. [cited 2023 Jan 15]. Available from: https://cv-tricks.com/keras/understand-implement-resnets/
80. Brownlee J (2018) How to use weight decay to reduce overfitting of neural network in keras [Internet]. [cited 2020 Aug 25]. Available from: https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/
81. Zhang X, Chen X, Yao L, Ge C, Dong M (2019) Deep neural network hyperparameter optimization with orthogonal array tuning. ICONIP 2019. 2019 Dec 12-15; Sydney. Berlin: Springer
82. Welch G, Bishop G (2006) An introduction to the Kalman filter [Internet]. 2006 Jul 24 [cited 2023 Jan 15]. Available from: https://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf
83. Lee DT (1983) On finding the convex hull of a simple polygon. Int. J. Comput. Inf 12:87–98. Available from: https://doi.org/10.1007/BF00993195
84. WOLFRAM Demonstrations project. Convex Hull [Internet]. 2007 Sep 28 [cited 2023 Jan 15]. Available from: https://demonstrations.wolfram.com/ConvexHull/
85. Sadli R (2020) Object tracking: 2-D object tracking using Kalman filter in Python [Internet]. [cited 2023 Jan 15]. Available from: https://machinelearningspace.com/2d-object-tracking-using-kalman-filter/
86. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. CVPR 2016. 2016 Jun 27–30; Las Vegas. New York: IEEE
87. Borovec J, Abdulla W. inspect_data.ipynb [Internet]. 2019 Feb 25 [cited 2023 Jan 15]. Available from: https://github.com/matterport/Mask_RCNN/blob/master/samples/coco/inspect_data.ipynb
88. Cheng T, Wang X, Chen S, Zhang W, et al. (2022) Sparse instance activation for real-time instance segmentation. CVPR 2022. 2022 Jun 18–24; New Orleans. New York: IEEE
89. Wang X, Zhang R, Kong T, Li L, et al. (2020) SOLOv2: dynamic and fast instance segmentation. NeurIPS 2020. 2020 Dec 6–12; Vancouver. La Jolla: Neural Information Processing Systems Foundation
90. Cao J, Anwer RM, Cholakkal H, Khan F, et al. (2020) SipMask: spatial information preservation for fast image and video instance segmentation. ECCV 2020. 2020 Aug 23-28; Glasgow. Berlin: Springer