International Journal of Computer Integrated Manufacturing

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tcim20

Multiple-sensor fault detection and isolation using video processing in production lines

Ali Abdo , Jamal Siam , Bashir Salah & Mohammed Krid

To cite this article: Ali Abdo , Jamal Siam , Bashir Salah & Mohammed Krid (2020) Multiple-sensor fault detection and isolation using video processing in production lines, International Journal of Computer Integrated Manufacturing, 33:6, 531-549, DOI: 10.1080/0951192X.2019.1667540

To link to this article: https://doi.org/10.1080/0951192X.2019.1667540

Published online: 01 Oct 2019.

INTERNATIONAL JOURNAL OF COMPUTER INTEGRATED MANUFACTURING
2020, VOL. 33, NO. 6, 531–549
https://doi.org/10.1080/0951192X.2019.1667540

ARTICLE

Multiple-sensor fault detection and isolation using video processing in production lines

Ali Abdo [a], Jamal Siam [a], Bashir Salah [b,c] and Mohammed Krid [b]

[a]Electrical and Computer Engineering Department, Faculty of Engineering and Technology, Birzeit University, Ramallah, Palestine; [b]Industrial Engineering Department, College of Engineering, King Saud University, Riyadh, Saudi Arabia; [c]Advanced Manufacturing Institute, King Saud University, Riyadh, Saudi Arabia

ABSTRACT
Production-line sensors are essential for synchronisation and execution of the correct workflow in a manufacturing process. Usually, sensor faults lead to serious delays, or even the complete termination of the manufacturing process, to carry out maintenance. Thus, sensor faults compromise the reliability of the manufacturing system and disturb the production schedule. This study proposes a multiple-sensor fault detection and isolation scheme for manufacturing series production lines. The proposed scheme adopts a global-redundancy method, using a digital camera and an ad-hoc video processing algorithm to detect and isolate faulty sensors. The main objective of this research is to preserve continuity of the production workflow and solve the problem of production delays and interruptions. Moreover, the scheme provides the possibility of online and post-process system maintenance. Further, the collected information on sensor false alarm rates can be used for a reliability analysis of the production line. The proposed scheme was tested using a laboratory production line model. The results show that the proposed scheme achieves the established objectives and improves the reliability of the manufacturing process.

ARTICLE HISTORY
Received 17 December 2018
Accepted 22 August 2019

KEYWORDS
Fault detection; fault isolation; video/image processing; fault-tolerant control; production lines; finite state machine

1. Introduction

The fusion of different technologies across all human disciplines is a futuristic trend that characterises the fourth industrial revolution (Schwab 2017). An example of this revolution is the merging of image processing and video-based monitoring techniques in various fields, such as industrial and medical applications (Aiteanu, Ristic, and Graser 2005; Abdo et al. 2015; Shleymovich, Medvedev, and Lyasheva 2016; Jacobsen and Ott 2017; Lee and Yang 2017).

An analysis of the literature reveals an increased use of data models and visual data in system modelling and analysis, and special attention is now being given to video and image processing techniques and their applications in automatic control. Previously, video and image processing of visual data was mainly used for system monitoring. Recent studies have fostered the importance of visual information in process control. For example, cameras are now used to provide visual and non-contact measurements of various control schemes (Liu et al. 2006; Tombari et al. 2008; Wang, Liao, and Mei 2008; Wang et al. 2010; Janabi-Sharifi and Marey 2010; Grigorescu and Moldoveanu 2011; Wang, Gao, and Qiu 2016). In addition, visual information analysis techniques have been used in various applications, such as stability and control design (Grigorescu and Moldoveanu 2011; Wang, Liao, and Mei 2008; Murao, Kawal, and Fujita 2009).

Image and video processing techniques can also play an important role in fault detection and fault-tolerant system design. Fault diagnosis (FD) and fault-tolerant control (FTC) are two processes that aim to improve the reliability, availability, dependability, and safety of systems. Several research studies have implemented FD and FTC for this purpose (Chen and Patton 1999; Mahmoud, Jiang, and Zhang 2003; Isermann 2006; Gao and Ding 2007; Blanke et al. 2008; Noura et al. 2009; Abdo et al. 2012a, 2012b; Ding 2014; Abdo, Siam, and Al-Rimawi 2017; Prieto, Duran, and Barrero 2017).

Fault diagnosis is a general concept and includes three essential tasks: (1) detection, which

CONTACT Ali Abdo aabdo@birzeit.edu Electrical and Computer Engineering Department, Faculty of Engineering and Technology, Birzeit University, Ramallah, Palestine.

detects the occurrence of a system fault, (2) fault isolation, which classifies the faults into predefined categories, and (3) fault identification, which determines the fault parameters, such as magnitude, type, and source. Fault diagnosis can be achieved using different techniques, such as hardware redundancy, plausibility testing, signal processing, and model-based fault detection (software redundancy) (Ding 2013; Zhang and Jiang 2008; Chen and Patton 1999; Isermann 2006).

Several methods have been employed to solve fault detection and isolation problems. Demetgul implemented two artificial neural networks for fault detection in a didactic modular production system produced by Festo Company (Demetgul, Tansel, and Taskin 2009). Qiao proposed a novel method for the diagnosis of rotating machinery based on an improved wavelet package transform (Qiao et al. 2007). Zhang used genetic programming to detect faults in rotating machinery (Zhang, Jack, and Nandi 2005). Widodo proposed a machine-learning algorithm based on principle a component analysis and support vector machine for induction motor diagnosis (Widodo, Yang, and Han 2007). Szkilnyk proposed a machine vision system that detects and isolates faults based on the spatiotemporal distribution of events during normal and abnormal operation (Szkilnyk et al. 2012). Chauhan developed a machine vision system using three different detection methods: Gaussian Mixture Model, optical flow, and running average (Chauhan and Surgenor 2015).

The global performance of a system is improved through the integration of FTC in plant systems and system controllers, such as electrical drives, vehicles, and wind turbines (Liu, Saif, and Fan 2016; Yu, Li, and Zhang 2018; Li et al. 2018). FTC can be accomplished using passive and active schemes. Passive FTC systems operate under a control law that is designed to be robust against fault occurrence. On the other hand, active FTC systems require reconfiguration of the control law after a fault occurrence.

Although important research has been conducted pertaining to FTC, the use of visual information and FTC in industrial production lines and manufacturing processes warrants further investigation. A literature review revealed that some studies have introduced the use of digital cameras and FTC schemes in industrial processes. Wang, Gao, and Qiu (2016) discussed the use of FTC in an industrial process based on predictive control. Van et al. (2016) developed an FD system for a robot using image processing. Lee and Yang (2017) used a smart industrial camera to perform object recognition and counting. Recently, Abdo, Siam, and Al-Rimawi (2017) proposed the integration of camera-based visual monitoring in an FTC scheme. Despite the aforementioned research, the use of digital cameras and video processing in FTC has not been comprehensively investigated. This study proposes the integration of a digital camera and an ad-hoc video processing algorithm with a production line manufacturing process to achieve a multiple-sensor FTC scheme. The use of the digital camera aims to provide a global redundancy to the multiple sensors in the production line. The video processing algorithm is employed to detect and isolate the faulty sensors. The main objective of FTC is to ensure the continuity of the production line workflow and the correctness of the manufacturing action synchronisation in the presence of object-detection sensor faults. Other issues, such as those related to actuator faults and processed-object characteristics or quality, are not included in this study. The proposed FTC scheme is tested using a laboratory-scale production line model. The results showed that the proposed scheme achieved its objectives and improved the reliability of the manufacturing process. It is believed that the modularity of the FTC structure and its relative processing algorithm ensure that the proposed scheme can be transferred to real industrial production lines. Thus, the next phase of this study is the application of the proposed FTC to a real industrial production line. Additional future work, such as the use of digital cameras, video processing, and computer vision, is required to consider product-quality sensor faults and actuators faults.

To the best of the authors' knowledge, a similar FTC scheme has not been previously proposed in the literature. The major contributions of this study are as follows:

- Introduction of a sensor global-redundancy scheme.
- Development of a simple and efficient technique for multiple-sensor fault detection and isolation based on video and image processing.
- Improvement of the production line monitoring system and manufacturing process workflow by providing alternative signals to isolate the faulty-sensor measurements.
- Prevention of any production line and manufacturing process interruptions.
- Allowance of online and post-process maintenance of the production line sensors.
- Evaluation of the reliability of the production line sensors.
- Improvement of the global safety and reliability of the production line (i.e. the avoidance of human injury, machine damage, and production deterioration).

The remainder of the paper is organised as follows: Section 2 presents the basic idea and development of the problem formulation for the proposed scheme using a real case study of an automatic production line. The image-processing algorithm and its outcomes are presented in Section 3. FTC is defined in Section 4. Section 5 presents the experimental results of different scenarios using a laboratory-scale production line. Finally, Section 6 illustrates the concluding remarks of this study.

2. Problem formulation

2.1. Basic idea

To illustrate the problem of sensor faults, this study uses a non-buffered series production line (as shown in Figure 1), where several objects are processed in parallel; when the action at one operation site terminates, the object is passed to the next site. Therefore, object placement at the various operation sites must be synchronised with the processing actions. This synchronisation is generally achieved using object-detection sensors. The proposed production line consists of a conveyor belt, a direct current (DC) motor, a DC motor electronic-drive circuit, a set of object-detection

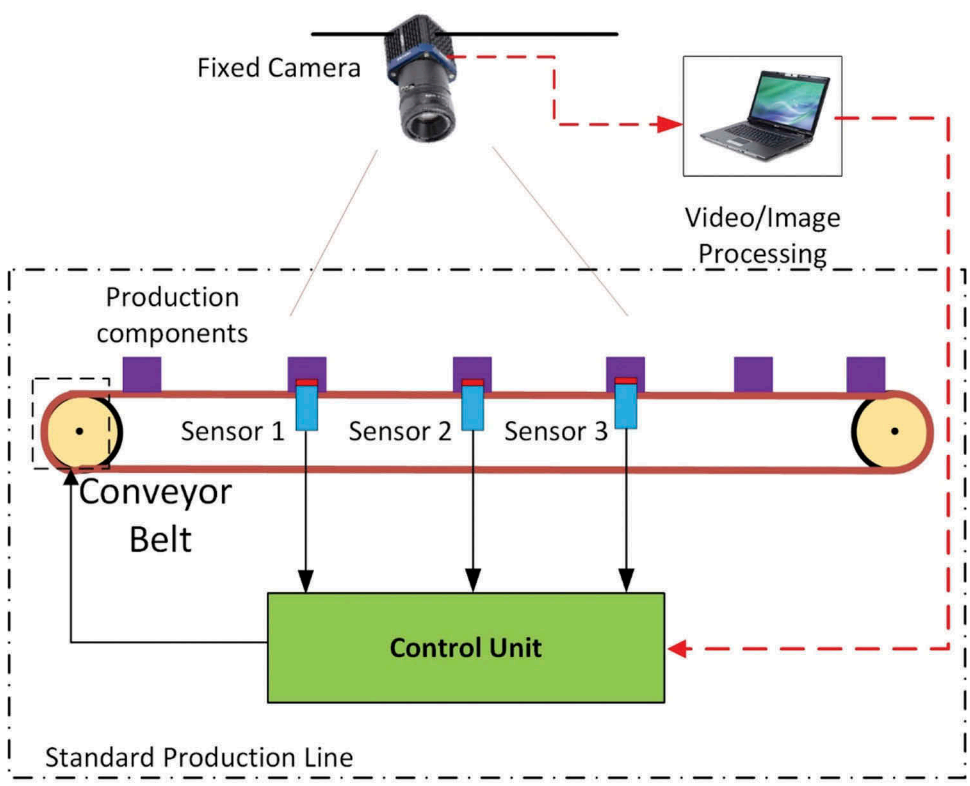

Figure 1. The production line configuration used to illustrate the proposed FTC. The configuration includes several operation sites with object-detection sensors and a series operation scheme.

sensors, a set of functional units, and an electronic digital controller. The DC drive, functional-unit inputs, and sensor outputs are assumed to be digital (on/off). In the absence of sensor faults, the workflow assumes that the object positioning, transport, and action of the functional units are controlled by the sensor feedback measurements. Thus, the objects under process (OUPs) are injected onto the conveyor belt and are transported until they are detected by the sensors at the operation site. Consequently, the conveyor belt is stopped for a fixed, predefined period (action expected time), during which the OUPs are processed by the functional unit. The control unit then generates the control signals that drive the motor and transport the OUPs to the next position.

A simple investigation of the production line workflow reveals that the synchronisation of the manufacturing process is strictly related to the functionality of the object-detection sensors. Thus, any sensor fault will result in a production line synchronisation failure and manufacturing workflow disruption. For example, consider the case in which one or more sensors fail to detect the OUP. The input of the control unit relative to these sensors will remain deactivated, and the conveyor belt will continue moving. Consequently, the OUP is not processed at that operation site, resulting in a manufacturing error. Additionally, another failure scenario can occur if a sensor fault activates the input of the control unit when an OUP is not present at the operation site. In this case, the functional-unit action is executed in the absence of an OUP.

To avoid these and other similar manufacturing failures, object-detection sensor faults must be detected and isolated in real time to avoid downtime of the production line for maintenance. In summary, object-detection sensor faults have serious critical consequences on the manufacturing process, such as the production of defective components, delays in the production schedule, and increases in production cost. Therefore, it is essential to compensate the effects of the faulty sensors and avoid downtime of the production line for maintenance in order to enhance the tolerance, reliability, and performance of the manufacturing process.

Sensor redundancy is a plausible solution that may avoid the adverse effects of sensor faults. The use of two or more sensors to detect the presence of objects at operation sites is termed hardware redundancy. Herein, the inclusion of redundant sensors to measure a specific variable is referred to as local redundancy. However, the implementation of local redundancy schemes in manufacturing processes is expensive due to the large number of sensors required. Therefore, a global-redundancy scheme can be more practical. Global redundancy uses a single detector to provide information pertaining to a set of variables. This study employs a global-redundancy scheme by using a digital video camera, as indicated in Figure 1, and a video processing algorithm to provide information about the presence/absence of the OUPs at the operation sites. The digital camera captures a sequence of frames. These frames include information about the motion of the OUPs and their relative distances with respect to the operation sites. The video processing algorithm extracts the OUP information, detects the presence/absence of the OUPs at the operation sites, and generates detection signals and fault identification codes. The control unit uses this information and the FTC rules to compensate the effects of the faulty sensors. Thus, the proposed FTC system ensures the continuity of the manufacturing process sequences and prevents production line downtime.

2.2. Problem statement

To accomplish the proposed multiple-sensor FTC scheme, the following tasks are required:

- Acquisition of video frames.
- Application of video and image processing techniques to detect the OUP positions.
- Construction of the FTC rules that consider the sensors and camera-based signals.
- Application of the FTC rules, which will compensate the effects of the sensor faults in real time to avoid workflow interruption and production errors.
- Identification of the faulty sensors and generation of alarm signals for the purposes of online or post-process maintenance.

3. Proposed architecture

The camera is assumed to be fixed and is positioned to provide a visual that covers all locations that require monitoring to control the manufacturing process. The aim is to associate a region-specific image to the operation site of each functional unit. The camera is connected to a data acquisition and processing unit. This unit implements the video/image processing algorithm and generates the redundant sensing signals. The visual data is managed in four phases: video acquisition, video handling, image processing, and fault evaluation. The video frame rate is set to the minimum rate, FR_{min}, which suits the dynamic of the manufacturing process. The minimum frame rate is determined by Equation (1), where v_{conv} is the conveyor belt speed in m/s, and D_{min} is the OUP desired position resolution.

$$FR_{min} = \frac{v_{conv}}{D_{min}} \quad (1)$$

The video frames are acquired as a sequence of images with a defined time tag. Hence, the amount of time an object is present in a given region of can be computed using the frame tags and the total number of frames that are acquired during the presence of the object in the region. This presence time is then used to detect the occurrence of sensor faults by comparing the computed presence time and the expected presence detection time. The expected presence detection time is assumed to be known from the normal operation timing of the production line. Consequently, the processing algorithm identifies the faulty sensors and isolates them by generating a set of redundant signals to replace the faulty-sensor measurements. These signals are then transferred to a control unit, such as a programmable logic controller (PLC), to maintain the correct workflow of the production line.

4. Fault-tolerant control

The proposed FTC is shown in Figure 2 and is composed of the production line, digital camera, and processing and control units. It should be noted that the proposed FTC does not introduce any changes to the original system settings; the proposed FTC adds a set of redundant monitoring signals that must be executed by the

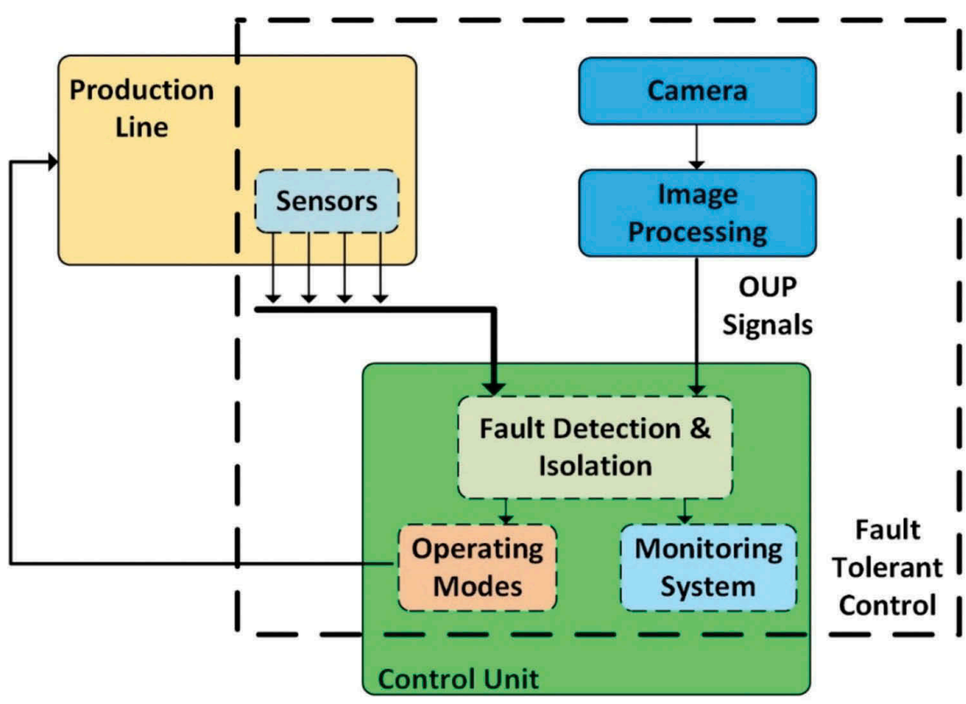

Figure 2. Proposed FTC scheme.

control algorithm. Moreover, the FTC scheme does not impose any additional conditions on the manufacturing process workflow. The processing algorithm augments the original system with new features, such as reliability analysis and online maintenance, without additional requirements to the original system parameters. The FTC scheme assumes that the digital camera is fault free, since it generates the reference signal.

4.1. Operation modes

A finite state machine is a logical way to represent a set of rules and conditions and to handle various inputs. Industrial processes are commonly represented by finite state machines. Therefore, to illustrate and validate the proposed FTC scheme, the FD-FTC system is modelled as a finite state machine, as illustrated in Figure 3. The various operation modes and the state-flow transitions were simulated and tested using MATLAB/Simulink. These modes are described as follows:

Operation mode 1: normal operation

In this case, the production line sensors correctly detect the presence of the OUPs in their relative operation sites. The control unit sets the normal operation flag (NOF) and operates using the sensors signals.

Operation mode 2: OUP missed detection

In this case, one or more sensors fail to detect the presence of the OUPs. However, the object presence is detected by Algorithm 2, which generates the object presence signal (OPS) and the faulty-sensor identification codes. The signals and codes are then transferred to the control unit (Algorithm 1), which isolates the faulty sensors and generates a missed detection flag (MDF). The OPS signals are used to maintain the workflow of the manufacturing process.

Operation mode 3: OUP false alarm

In this case, the sensors erroneously indicate the presence of one or more OUPs. That is, the sensors detect the presence of the OUPs that are absent. The image processing algorithm detects the absence of the OUPs and generates the required OPS values and fault identification codes. The information is sent to Algorithm 1, which isolates the faulty sensors and generates a false alarm flag (FAF).

Operation mode 4: OUP premature motion

In this case, the object-detection sensor indicates the absence of the OUP before the termination of the expected period of action. Thus, the object is moved from the operation site prior to completion of the processing action, which is known as premature object motion (POM). Algorithm 2 detects this anomaly and generates a POM signal and the faulty-sensor identification code. This information is then sent to Algorithm 1, which isolates the faulty sensor and generates a premature motion flag (PMF).

Operation mode 5: OUP long presence

In this case, the sensor fault indicates that the OUP is present after the expected period of action is terminated (i.e. the object has not moved from the operation site at the designated time). This condition is detected by Algorithm 2, which generates a long presence signal (LPS) and the fault identification code. Algorithm 1 isolates the faulty sensor and generates a long presence flag (LPF).

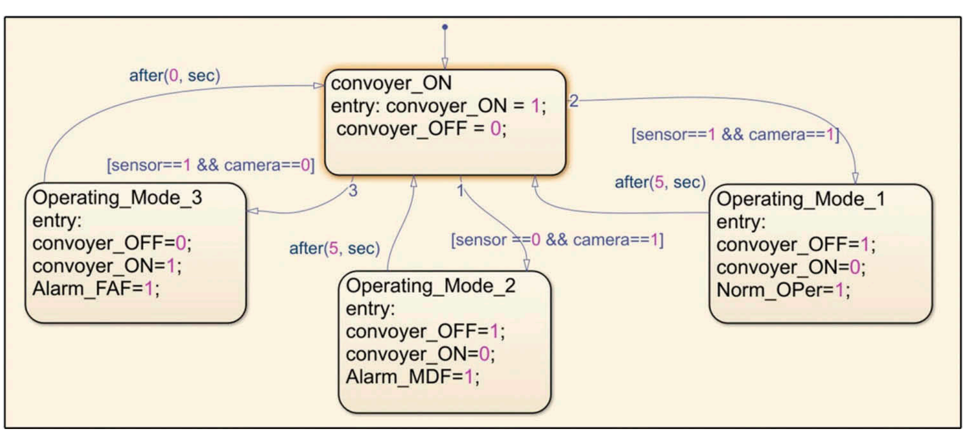

Figure 3. Finite state machine for the normal, missed-detection, and false alarm modes.

4.2. Additional features

4.2.1. Sensor reliability analysis

The FTC system ensures the continuity of the manufacturing process, regardless of sensor faults. It detects the existence of faulty sensors that contain persistent or intermittent faults. For persistent faults, the identified faulty sensors require immediate maintenance. However, in the case of intermittent faults, there may be difficulties in locating and identifying the faulty sensors because the operation of the production line is restored when the sensor fault vanishes. In the proposed FTC scheme, the number and frequency of sensor faults are generated by the image-processing algorithm. The data collected during one or multiple manufacturing cycles can be used to analyse the reliability of the sensors. Moreover, since the sensor reliability affects the production line operations, a sensor reliability analysis can be used to improve the reliability of the whole production line. In fact, decisions regarding sensor maintenance or replacement can be made based on the sensor reliability reports.

4.2.2. Online maintenance

In the case of persistent sensor faults, the FTC ensures the continuity of the production line operation by the OPSs. The FTC flags (MDFs and FAFs) identify the faulty sensors and designate a maintenance-request flag. Thus, the maintenance team can apply ad-hoc online maintenance procedures to restore the sensing functions or schedule a post-operation offline maintenance procedure.

4.3. Algorithms

This subsection introduces the process control algorithm, which implements the proposed FTC scheme, as well as the fault-detection video and image algorithm.

4.3.1. Process control algorithm

The basic steps of the process control algorithm (Algorithm 1) that apply the FTC scheme on a total number (total_objects) of OUPs (Object_Under_Process) are as follows:

- If the number of objects is less than the total number of objects, a new OUP is injected onto the conveyor belt (conveyor_belt), which enables the movement of the OUP to the operating site.
- If the object is not detected by the sensor, Algorithm 1 receives OPS from the video processing algorithm (Algorithm 2). Conveyor_belt is then stopped with the OUP at the operation site, and the functional unit is activated. In this case an MDF is issued by the control unit. If fault_counter is larger than a given threshold (fault_maintenance_threshold), a maintenance flag (MNTF) is also issued.
- If Algorithm 1 receives a POM signal from Algorithm 2, this indicates that the OUP began movement before the termination of the designated action expected period (action_expected_period). Then, Algorithm 1 forces conveyor_belt to cease keeping the OUP at the operation site. In this case, a PMF is issued. In addition, if fault_counter is higher than the designated threshold, an MNTF is issued by Algorithm 1.
- If Algorithm 1 receives an LPS from Algorithm 2, this indicates that the OUP was not released from the operation site after the termination of action_expected_period. Then, conveyor_belt is forced to move the object to the following operation site. In this case, Long_Presence_Flag (LPF) is issued, and if fault_counter is higher than the designated threshold, an MNTF is issued by Algorithm 1.
- If the sensor signal (sensor_signal) indicates the presence of an object, Algorithm 1 receives an OPS value from Algorithm 2, indicating the absence of an object at the operation site. Algorithm 1 forces conveyor_belt to move until an OPS is received from Algorithm 2, which indicates the arrival of the signal to the operation site. Consequently, Algorithm 1 positions the OUP at the operating site, and the operation continues as expected under control of the proposed FTC scheme. In this case, an FAF is issued. In addition, if fault_counter is higher than the designated threshold, an MNTF is issued by Algorithm 1.
- If sensor_signal detects the presence of the OUP and Algorithm 1 receives an OPS that indicates the same condition, then the action follows normal operation mode, and the operation continues as expected with the control of the sensor. In this case, Normal_Operation_Flag is issued by Algorithm 1. However, if Algorithm 1 receives a POM or an LPS from Algorithm 2, the abovementioned relative procedures are applied, and the process continues as expected.
- The procedure is repeated until all OUPs are processed, as designated by total_objects.

Table 1. List of symbols used in Algorithm 1.

symbol	meaning	symbol	meaning
OPS	Object_Presence_signal	NOF	Normal_Operation_Flag
POM	Premature_Object_Movement	MDF	Miss-Detection_Flag
LPS	Long_Presence_signal	FAF	False_Alarm_Flag
OUP	Object_Under_Process	PMF	Premature_Motion_Flag
LPF	Long_Presence_Flag	MNTF	Maintenance_Flag

Algorithm 1: Process Control Algorithm

Algorithm 1:
Input: OPS, POM, LPS, OUP, sensor_identification_code, sensor_signal
Output: NOF, MDF, FAF, PMF, LPF, MNTF
Parameters: total_objects, fault_maintenance_threshold

1: **begin**
2: object_count = 0
3: fault_counter = 0
4: **if** (**object_count > total_objects**) **then**
4.1: **goto 5.6.1**
5: **else**
5.1: **inject** OUP
5.2: **increment** object_count
5.3: **move** conveyor_belt
5.4: **read** input
5.5: **if** (**sensor_signal = = 0**) **then**
5.5.1: **if** (**OPS = = 0) then**
5.5.1.1 **deactivate** functional_unit
5.5.1.2 **goto 5.4**
5.5.1.3: **else if** (**OPS = = 1**) **then**
5.5.1.3.1: **stop** conveyor_belt
5.5.1.3.2: **activate** functional-unit
5.5.1.3.3: **out** MDF
5.5.1.3.4: **out** sensor_identification _code
5.5.1.3.5: **increment** fault_counter
5.5.1.3.6: **if** (**fault_counter > fault_maintenance_threshold**) **then**
5.5.1.3.6.1: **out** MNTF
5.5.1.3.7: **end**
5.5.1.3.8: **read** input
5.5.1.3.9: **if** (**OPS = = 1 && POM = = 1**) **then**
5.5.1.3.9.1: **activate** functional unit
5.5.1.3.9.2: **stop** conveyor_belt
5.5.1.3.9.3: **out** PMF
5.5.1.3.9.4: **out** sensor_identification _code
5.5.1.3.9.5: **increment** fault_counter
5.5.1.3.9.6: **if** (**fault_counter > fault_maintenance_threshold**)
5.5.1.3.9.6.1: **out** MNTF
5.5.1.3.9.7: **end**
5.5.1.3.9.8: **else if** (**OPS = = 0 && LPS = = 1**) **then**
5.5.1.3.9.8.1: **deactivate** functional unit

```
5.5.1.3.9.8.2:                                  move conveyor_belt
5.5.1.3.9.8.3:                                  out LPF
5.5.1.3.9.8.4:                                  out sensor_identification _code
5.5.1.3.9.8.5:                                  increment fault_counter
5.5.1.3.9.8.6:                                  if (fault_counter > fault_maintenance_threshold)
5.5.1.3.9.8.6.1:                                    out MNTF
5.5.1.3.9.8.7:                                   end
5.5.1.3.9.8.8:                                   else if (OPS = = 1 && POM = = 0 && LPS = 0) then
5.5.1.3.9.8.8.1:                                     goto 5.5.1.3.8
5.5.1.3.9.8.9:                                   end
5.5.1.3.9.8.10:                                  else if (OPS = = 0 && POM = = 0 && LPS = = 0) then
5.5.1.3.9.8.10.1:                                     goto 4
5.5.1.3.9.8.11:                                    end
5.5.1.3.9.9:                                    end
5.5.1.3.10:                                  end
5.5.1.4:                                 end
5.5.2:                           end
5.5.3                            else if (sensor = = 1) then
5.5.3.1                               if (OPS = = 0) then
5.5.3.1.1                                 deactivate functional-unit
5.5.3.1.2                                 out FAF
5.5.3.1.3                                 increment fault_counter
5.5.3.1.4                                 if (fault_counter > fault_maintenance_threshold) then
5.5.3.1.4.1                                   out MNTF
5.5.3.1.5                                 end
5.5.3.1.6                                 else if (OPS = = 1) then
5.5.3.1.6.1                                    stop conveyor_belt
5.5.3.1.6.2                                    activate functional_unit
5.5.3.1.6.3                                    out NOF
5.5.3.1.6.4                                    goto 5.5.1.3.8
5.5.3.1.7                                  end
5.5.3.2:                              end
5.5.4:                           end
5.6:    end
5.6.1:    move conveyor_belt
5.6.2:    eject last_object
5.6.3:    stop conveyor_belt
5.6.4:    out 'total_objects finished'
5.6.5:    store fault_counter
6:   end
```

Table 2. List of symbols used in Algorithm 2.

symbol	meaning	symbol	meaning
OPS	Object_Presence_signal	ROI	Region_Of_Interest
POM	Premature_Object_Movement	LPS	Long_Presence_signal

Algorithm 2: Fault-Detection Video and Image Processing Algorithm

Algorithm-2
Input: video_frame, frame_rate
Output: OPS, POM, LPS, fault_identification_code
Parameters: presence_detection_threshold, action_expected_period
1: **Begin**
2: **read** video_frame
3: **extract** ROI
4: **set** ROI_temp = ROI
5: **read** video_frame
6: **extract** ROI
7: **compute** background_reference = ROI-ROI_temp
8: **compute** BW_reference = convert (background_reference to blank and white)
9: **compute** BW_ref_average = average (BW_reference)
10: **set** frame_count = 0
11: **read** video_frame
12: **extrac**t ROI
13: **compute** background_difference = ROI-background_reference
14: **compute** BW_difference = convert (background_difference to blank and white)
15: **compute** BW_average = average (BW_difference)
16: **if** (**BW_average>BW_ref_average**) **then**
16.1: **increment** frame_count
16.2: **goto 11**
16.3: **else if** (**frame_count = = 0**) **then**
16.3.1: **goto 11**
16.3.2: **else**
16.3.2.1: **compute** presence_time = frame_count/frame_rate
16.3.2.2: **if** (**presence_time ≥ presence_detection_threshold**) **then**
16:3.2.2.1: **set** OPS = 1
16:3.2.2.2: **out** OPS
16:3.2.2.3: **read** video_frame
16:3.2.2.4: **extrac**t ROI
16:3.2.2.5: **compute** background_difference = ROI-background_reference
16:3.2.2.6: **compute** BW_difference = convert (background_difference to blank and white)
16:3.2.2.7: **compute** BW_average = average (BW_difference)
16:3.2.2.8: **if** (**BW_average>BW_ref_average**) **then**
16:3.2.2.8.1: **increment** frame_count
16:3.2.2.8.2: **goto 16:3.2.2.3**
16:3.2.2.8.3: **else**
16:3.2.2.8.4: **compute** presence_time = frame_count/frame_rate
16:3.2.2.8.5: **if** (**presence_time = action_expected_period**) **then**
16:3.2.2.8.5.1: **set** OPS = 0
16:3.2.2.8.5.2: **out** OPS
16:3.2.2.8.5.3: **else if** (**presence_time < action_expected_period**) **then**
16:3.2.2.8.5.3.1: **set** POM = 1
16:3.2.2.8.5.3.2: **out** POM
16:3.2.2.8.5.3.3: **set** OPS = 1
16:3.2.2.8.5.3.4: **out** OPS

```
16:3.2.2.8.5.3.5:                                    generate fault_identification_code
16:3.2.2.8.5.3.6:                                    out fault_identification_code
16:3.2.2.8.5.3.5:                                    else if (presence_time > action_expected_period) then
16:3.2.2.8.5.3.5.1:                                        set OPS = 0
16:3.2.2.8.5.3.5.2:                                        out OPS
16:3.2.2.8.5.3.5.3:                                        set LPS = 1
16:3.2.2.8.5.3.5.4:                                        out OPS
16:3.2.2.8.5.3.5.5:                                        generate fault_identification_code
16:3.2.2.8.5.3.5.6:                                        out fault_identification_code
16:3.2.2.8.5.3.6:                                      end
16:3.2.2.8.5.4:                                      end
16:3.2.2.8.6:                            end
16:3.2.2.9:                          end
16:3.2.3:                        end
16:3.3                       end
16:4        end
16:5        goto 10
17:     end
```

4.3.2. Fault-detection video and image processing algorithm

Algorithm 2 describes the image-processing algorithm. This algorithm uses the digital camera and Algorithm 1 to achieve the global redundancy and implement the FTC scheme. Algorithm 2 detects the sequence of OUP presence/absence at the operation sites in accordance with the planned manufacturing workflow and timing. The algorithm generates OPS, POM, and LPS signals and fault_identification_codes. These are sent to Algorithm 1 to control the processing sequences. The basic steps of the algorithm are as follows:

- The algorithm starts by acquiring a sequence of video frames in the absence of OUPs. The algorithm then extracts a small region of interest (ROI). The ROI is used to detect the presence/absence of an OUP at the operation site.
- To generate reference_background images, two video frames are acquired and subtracted. Reference_background is then converted to a black and white format (BW_reference), and its average value (BW_ref_average) is computed and used as a threshold to determine the presence/absence of the OUP.
- Video frames are continuously acquired during the production line manufacturing process. The ROI of each frame is extracted. The average value of the difference between the ROI and reference_background in the black and white form is computed (BW_average). The presence of an object is detected if BW_average is larger than BW_ref_average. Additionally, presence_time is computed from the consecutive sequence of the ROI that satisfies the presence condition.
- If the computed presence time is larger than a previously determined threshold (presence_detection_threshold), which is dependent upon the speed of conveyor_belt and the minimum processing time, then the OUP is considered correctly positioned at the operation site. Consequently, the OPS is set to 1 and sent to Algorithm 1.
- After determining the correct positioning condition, Algorithm 2 continues to read new video frames. Two anomalies may occur due to sensor faults. In the first anomaly, the OUP may leave the operation site before action_expected_period, which is previously determined and depends on the operation executed by the functional unit. Algorithm 2 sets POM to 1 and OPS to 1 and sends them to Algorithm 1. In the second anomaly, the OUP may remain at the operation site for a time longer than action_expected_period. Algorithm 2 sets OPS to 0 and LPS to 1 and sends them to Algorithm 1. In both cases, Algorithm 2 generates and sends the fault

identification code, which indicates the operation site of the faulty sensor that requires isolation.

- Algorithm 2 operates indefinitely and can be stopped when all total_objects are processed.

4.3.3. Image processing algorithm: basic mathematical operations

This section provides an overview of the image processing operations used in Algorithm 2.

4.3.3.1. Video-frame representation.

The kth frame image of a video stream is a matrix of size M × N and is defined by Equation (2):

$$X_k = f_k(x_i, y_i) \tag{2}$$

with $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$.

ROI extraction. The r^{th} ROI of frame k is a matrix of size $(m_r - i_r) \times (n_r - j_r)$ and is defined by Equation (3):

$$W_{kr} = X_k(x_i, y_i) \tag{3}$$

with $1 \le i = i_r, i_r + 1, \ldots, m_r \le M,$ and $1 \le j = j_r, j_r + 1, \ldots, n_r \le N.$

RGB colour image to grey image conversion. Assume that $S = f(R_{i,j}, G_{i,j}, B_{i,j})$ is an RGB colour image. The associated grey image is obtained by Equation (4):

$$gray_{i,j} = (0.2989 * R_{i,j} + 0.5870 * G_{i,j} + 0.1140 * B_{i,j}) \tag{4}$$

Grey image to black and white image conversion. The grey image is represented by Equation (5):

$$X = f(x_i, y_i) \tag{5}$$

where, $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$ is a given grey image. The corresponding black and white image is defined by Equation (6):

$$\Pi i, j = \begin{cases} 0 & f(xi, yi) \le Th \\ 1 & f(xi, yi) > Th \end{cases} \tag{6}$$

Difference image computation. Assume that $f(x_i, y_i)$is a reference image, and $g(x_i, y_i)$ is an image of the same type and size as f. The difference image, $D(x_i, y_i)$, is equal to Equation (7).

$$f(x_i, y_i) = g(x_i, y_i) - f(x_i, y_i) \tag{7}$$

Average value of a black and white image computation. Assume a black and white image is represented by Equation (8).

$$X = f(x_i, y_i) \tag{8}$$

where, $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$ is a given black and white image. The average value of the image is defined by Equation (9).

$$X_{average} = \frac{1}{M \times N} \sum_{j=1}^{N} \sum_{i=1}^{M} f(x_i, y_i) \tag{9}$$

5. Experimental results

The proposed FTC system was tested in a laboratory using a real production line model, as shown in Figure 4. Tests were performed to examine and validate the applicability of the proposed scheme in real working scenarios. Several experiments were developed to test the responses to different sensor-fault conditions. The following subsections explain the different experimental setups and fault scenarios.

5.1. Experimental equipment and setup

The original system was represented by the production line model shown in Figure 4. The model consisted of a conveyor belt, a DC motor, three proximity inductive sensors, and a PLC. To focus on the experimental objective of sensor faults, the possibility of actuator faults must be excluded. Thus, each functional unit was represented by a light-emitting diode (LED). The LEDs were switched on and off according to the operating conditions throughout the predefined operation period (action expected period). The OUPs were small soft drink cans.

To implement the proposed FTC scheme, a digital video camera (FLIR BFS-U3-51S5C-C) was added to the system. The video was acquired at a rate of 30 fps. The ROI was defined as a rectangular area, which represents the back edge of an appropriately-positioned can during normal operating conditions. This was done to allow the control unit a sufficient amount of time to position the can at the operation site in case of a sensor fault. Each of the following experiments was repeated several times to ensure consistent outcomes.

5.2. Experiment 1: normal operation

In Experiment 1, a single OUP was injected onto the conveyor belt and was passed in front of the sensors. All of the sensors detected the presence of the OUP when it arrived at each respective operation

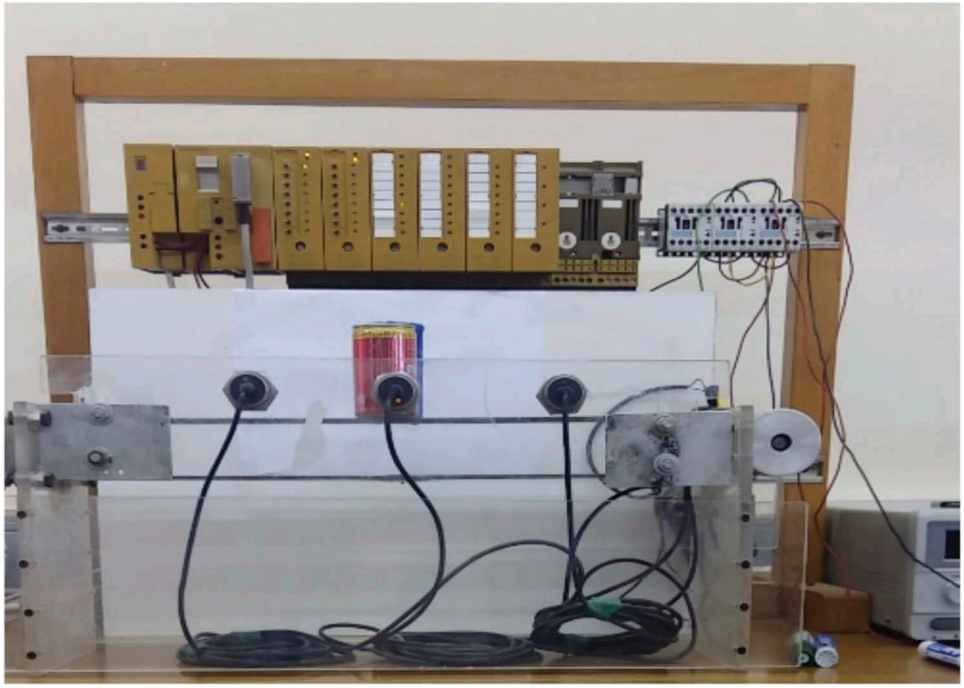

Figure 4. Laboratory test-bed settings with a real production line model.

Figure 5. ROI images (a) in the absence of an OUP and (b) in the presence of an OUP.

site. The ROI images that show the presence/absence of an OUP in the detection region (Algorithm 2) are shown in Figure 5. The ROI image sequences during the OUP movement into and out of the detection region are shown in Figures 6 and 7, respectively. The three functional-unit LEDs switched on and off, and the process operated as expected (Algorithm 1). Figure 7 shows that, under normal operation, the controller stopped the belt for the expected time period. The object then started to move out of the ROI after 150 frames (approximately 5 s) and left the ROI completely after 200 frames (approximately 6.7 s).

5.3. Experiment 2: single-sensor missed-detection fault

In Experiment 2, a single OUP was injected onto the conveyor belt and passed in front of the sensors, while excluding the proposed FTC scheme. To emulate a missed-detection condition, the second sensor signal was masked by the PLC program. The object passed in front of the masked sensor, and the conveyor belt continued to move without positioning the object at the operation site. The corresponding functional-unit LED was not switched on by the PLC control program, indicating a manufacturing failure. The results of the image-processing algorithm are shown in Figure 8. This figure shows that the OUP started to move out of the ROI 30 frames following its entry (approximately 1 s), and the OUP left the ROI completely after 30 frames (approximately 1 s) without stopping at the operation site.

The experiment was repeated while including the proposed FTC. When comparing the computed and expected presence times of the OUPs, Algorithm 2 classified the sequence of images in Figure 7 as a sensor 2 missed-detection fault. Accordingly, an OPS and a sensor 2 fault identification code were generated. The OPS was

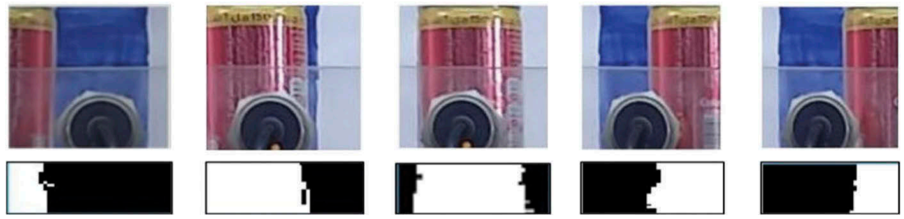

Figure 6. The sequence of extracted ROI images during the motion of the can into and out of the detection region (from left to right).

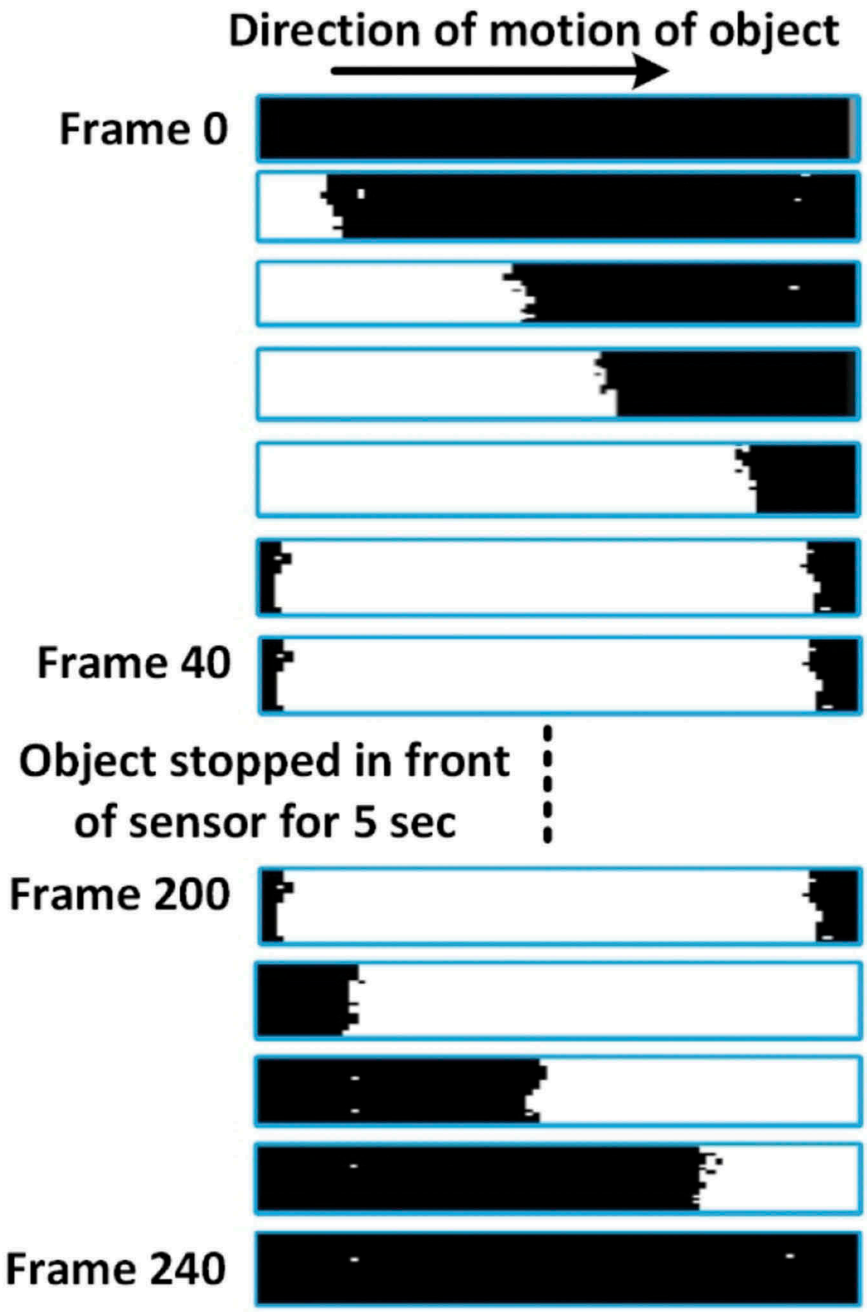

Figure 7. The frames tags and the sequence of images of the ROI during the object motion in normal operating mode.

Figure 8. The frame tags and sequence of the ROI images in the missed-detection mode without intervention of the proposed FTC. The object continued to move and was not positioned at the operation site.

read successfully by the PLC, the conveyor belt was stopped with the can at the operation site, and the functional-unit LED switched on. Following the completion of the action expected period, the LED switched off, and the conveyor belt movement was initiated. The OUP was detected at the next operation site, and the process operated as expected. The experiment was repeated for the first and third sensors, and similar results were obtained.

5.4. Experiment 3: multiple-sensor missed-detection fault

In Experiment 3, several OUPs were injected onto the conveyor belt, while excluding the proposed FTC scheme. Because the assumed operation is in series, the success of the manufacturing process requires the correct detection and placement of the cans at their respective operation sites. Therefore, the PLC was programmed to keep the conveyor belt moving when one or more sensors miss the OUP detection. The signals of the first and third sensors were masked by the control-unit program. The cans passed in front of the sensors, and the conveyor belt continued to move without positioning them at any operation site. An object was detected by the second sensor, and the second functional-unit LED switched on, indicating that it was operating, even though none of the cans were positioned at the second operation site. The first and third functional-unit LEDs were in the off state, indicating that none of the cans were detected at these sites.

The experiment was repeated with the intervention of the proposed FTC scheme. The image processing results at the three operation sites were similar to those of Figure 7. Algorithm 2 detected the fault conditions and generated OPS signals for the first and third operation sites. The sensor 1 and sensor 3 identification faults were also generated and sent to the control unit. The control-unit program stopped the conveyor belt with the cans correctly positioned and generated the MDFs. The functional-unit LEDs operated as expected.

The same experiment was repeated by masking sensors in various patterns: sensors 1 and 2, sensors 2 and 3, and sensors 1, 2, and 3. Similar results were obtained in all cases.

5.5. Experiment 4: false-alarm sensor fault

In the first portion of Experiment 4, a single can was injected onto the belt, while excluding the proposed FTC scheme. The signals of the second and third sensors were forced to permanently indicate the presence of cans. When the can arrived at the first operation site, the conveyor stopped, and the three functional-units LEDs switched on, indicating the presence of cans at the three operation sites. Thus, the processes at operation sites 2 and 3 were executed in the absence of cans, indicating a manufacturing failure.

The experiment was repeated with the inclusion of the proposed FTC scheme. The results showed that Algorithm 2 detected the presence of the can at the first operation site and the absence of the cans at the other sites. Consequently, the OPSs were deactivated, and the fault identification codes from sensors 2 and 3 were sent to the control unit. The control unit switched off the second and third functional-unit LEDs, whereas the LED of the first functional unit switched on, indicating the progress of the action at this site. The object was then moved to the following operation sites, was detected by Algorithm 2 and the forced sensor values, and generated the faulty-sensor codes. The control unit operated the production line correctly and issued the FAFs for operation sites 2 and 3.

The experiment was repeated for various faulty-sensor sites, and similar results were obtained.

5.6. Experiment 5: premature object movement

In this experiment, the cans were injected onto the conveyor belt without the new FTC scheme. The cans were correctly detected and positioned at the operation sites. Additionally, the LEDs switched on, indicating the progress of the action at their respective sites. Then, sensor 1 and sensor 2 were forced to indicate the absence of cans at these sites prior to the termination of the action expected period. The belt moved, removing the cans from of their operation sites, leading to a manufacturing failure.

The same sequence was repeated using the proposed FTC scheme. In this case, the premature motion of the cans was detected by Algorithm 2, and the POM signals were sent to the control unit. The control unit stopped the conveyor belt movement, and the operation sequence continued, as expected.

5.7. Experiment 6: long presence

In Experiment 6, the cans were injected onto the conveyor belt without the new FTC scheme. The cans were correctly detected and positioned at the operation sites; the LEDs switched on, indicating the progress of the action at their site. Then, all of the sensors were forced to indicate the presence of cans at their respective sites. At the termination of the action expected period, the LEDs switched off, indicating the termination of the functional-unit action. However, because the cans were detected again at the same operation sites, the belt did not move. The LEDs switched on, indicating the execution of the same actions on the same cans, leading to a manufacturing process failure.

The same sequence was repeated using the proposed FTC scheme. In this case, Algorithm 2 did not detect the release condition of the expected OUPs at the end of the action expected period. Accordingly, LPS signals and fault identification codes were generated and sent to the control unit. The control unit forced a motor switch on and restored the correct operation sequence of the production line.

6. Discussion

This study introduced a production line FTC system with a global-redundancy scheme to tolerate the misplacement of objects at the production line operation sites caused by object-detection sensor faults. The principle objectives were to ensure the synchronisation of processing actions and maintain the continuity of the manufacturing process workflow. Global redundancy was achieved using a video camera and a video processing algorithm. A series-type production line was assumed, and object-detection sensors were used to synchronise the OUP placement and functional-unit action at each operation site. The production line was assumed to operate without the use of buffers between the consecutive processing stages. Moreover, a fixed and equal action period was assumed for all the operation sites. The proposed FTC scheme was tested using a laboratory-scale small production line model. Finally, the objectives of the FTC scheme did not include sensors that monitor the quality of the products or possible actuator faults.

Despite these assumptions, the proposed FTC scheme has possible applications in real industrial production lines. In fact, several well-known production lines follow series manufacturing schemes. Some examples include automated food processing, metal processing, packing and filling systems. Although several series production lines are similar to the one proposed by the study, many others have buffering between the different processing stages. This buffering allows the different stages to operate asynchronously with the same operation time (balanced mode) or with different operation times (unbalanced mode). However, all cases require the OUP to be positioned at the operation site prior to executing the processing action. Thus, object-positioning errors caused by object-detection sensor faults can lead to manufacturing process failures in buffered and non-buffered production lines. This failure is instantaneous and holistic in non-buffered production lines; however, the failure is partial at the faulty site in buffered production lines. The persistence of a sensor fault at the faulty-sensor site leads to a delayed and complete manufacturing process failure of the entire buffered production line.

The proposed FTC was tested using a laboratory-scale production line, while controlling the fault conditions and scenarios. Moreover, the proposed object-detection algorithm was based on basic video and image processing operations. These operations included ROI extraction, image subtraction, thresholding, and a binary decision technique. However, the testing approach herein should not diminish the validity of the proposed scheme or its applicability in real industrial settings. In fact, similar testing methods were implemented in various literature studies (Demetgul, Tansel, and Taskin 2009; Sekar, Hsieh, and Wu 2011; Chauhan and Surgenor 2015).

In general, the most efficient and lowest cost solution is often the most successful, and the adoption of complex solution methods should be avoided. The object-detection algorithm proposed in this study achieved its objectives. A unique requirement of this study was the existence of a valid background reference. This requirement did not generate complications because the operation sites were assumed to be in fixed positions, which is also the case of various series production lines. Thus, the adoption of higher cost, more articulated, and more robust detection algorithms may be unsuitable and irrelevant when attempting to solve the proposed series production line problem. For example, Chauhan and Surgenor (2015) applied and compared three different machine vision methods to detect faults in an automated assembly machine. These methods included the Gaussian Mixture Model, optical flow, and a running-average approach using basic image processing operations. Their conclusion was that the simplicity of the running-average approach led to the best solution (Chauhan and Surgenor 2015).

It is worth noting that different settings must be implemented depending on the specific application of the series production line. In fact, the digital video camera and video processing microcomputer should be selected based on the specific timing and operating speeds of the production line. However, due to recent advances in digital cameras technology and processing systems, this should not be an obstacle to implementing the production line studied herein. Moreover, the timing of the entire production process and the local timing at each operation site in buffered mode should be taken into consideration. Due to the modular structure of the proposed FTC, this should not be a great challenge. In fact, several video cameras and processing units can be used to apply redundancy at different levels: global redundancy to the entire production line, sub-global redundancy to subsets of sensors, or local redundancy at each operation site. The generated signals and identification codes that are produced by the video processing algorithms must be sent to the production line control unit to perform the FTC actions.

The application and testing of the proposed FTC scheme in real industrial production line settings will be the subject of the following study. Other topics, such as product-quality sensors and actuator faults, may be studied in future works. The addition of these types of faults to the problem may require the employment of articulated algorithms that involve advanced video processing techniques, such as object tracking, machine learning, and computer vision.

7. Conclusion

This study proposed a new FTC scheme based on redundant monitoring through the use of a digital video camera and video processing. The proposed FTC aimed to foster the tolerance of the production line to multiple-sensor faults. The FTC system operating modes were simulated and tested using MATLAB/Simulink. To test the applicability of the proposed FTC scheme in practical working scenarios, several experiments were conducted using a laboratory production line model. The simulation and experimental results showed that the proposed FTC scheme achieved the desired objectives. The proposed scheme improved the tolerance and reliability of the production line to multiple-sensor faults. Moreover, the proposed FTC introduced other important features to the production line, such as the possibility of developing sensor and production line reliability analyses, and the application of online maintenance procedures.

Acknowledgments

The authors extend their appreciations to the Dean of Scientific Research at King Saud University for funding this work through research Group No (RG- 1439-003).

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the King Saud University [RG-1439-003].

ORCID

Ali Abdo http://orcid.org/0000-0001-6603-2126
Jamal Siam http://orcid.org/0000-0002-5962-8279
Bashir Salah http://orcid.org/0000-0001-5254-7698

References

Abdo, A., S. X. Ding, J. Saijai, and W. Damlakhi. 2012a. "Integration of Residual Evaluation and Threshold Computation into Switched Fault Detection System." Paper presented at the 8th IFAC Symposium SAFEPROCSS-2012, Mexico City, Mexico, August 29–31. doi:10.3182/20120829-3-MX-2028.00128.

Abdo, A., S. X. Ding, J. Saijai, and W. Damlakhi. 2012b. "Fault Detection for Switched Systems Based on Deterministic Method." Paper presented at 51th IEEE CDC, Maui, Hawaii, USA, December 10–13.

Abdo, A., J. Siam, D. Abu-obeid, and A. Awaineh. 2015. "Detection of Fractures and Hairline-fractures in X-Ray Images Using Image Processing." Paper presented at International Conference on Software, Multimedia and Communication Engineering, Hong Kong, September 20–21.

Abdo, A., J. Siam, and A. Al-Rimawi. 2017. "Visual Information Integration in Fault Tolerant Control System." Paper presented at 25th Mediterranean Conference on Control and Automation (MED), Malta, July 3–6.

Aiteanu, D., D. Ristic, and A. Graser, 2005. "Content Based Threshold Adaptation for Image Processing in Industrial Application." Paper presented at 5th International Conference on Control and Automation, Budapest, April 18–22.

Blanke, M., M. Kinnaert, J. Lunze, and M. Staroswiecki. 2008. *Diagnosis and Fault-Tolerant Control*. Berlin Heidelberg: Springer-Verlag.

Chauhan, V., and B. Surgenor. 2015. "A Comparative Study of Machine Vision Based Methods for Fault Detection in an Automated Assembly Machine." *Procedia Manufacturing* 1: 416–428. doi:10.1016/j.promfg.2015.09.051.

Chen, J., and R. Patton. 1999. Robust Model-Based Fault Diagnosis for Dynamic Systems. New York: Springer Science + Business Media, LLC.

Demetgul, D., I. N. Tansel, and S. Taskin. 2009. "Fault Diagnosis of Pneumatic Systems with Artificial Neural Network Algorithms." *Expert Systems with Applications* 36 (7): 10512–10519. doi:10.1016/j.eswa.2009.01.028.

Ding, S. X. 2013. *Model-based Fault Diagnosis Techniques: Design Schemes Algorithms, and Tools*. 2nd ed. Berlin Heidelberg: Springer-Verlag.

Ding, S. X. 2014. *Data-driven Design of Fault Diagnosis and Fault-tolerant Control Systems*. Berlin Heidelberg: Springer-Verlag.

Gao, Z., and S. X. Ding. 2007. "Actuator Fault Robust Estimation and Fault-tolerant Control for a Class of Nonlinear Descriptor Systems." *Automatica* 43 (5): 912–920. doi:10.1016/j.automatica.2006.11.018.

Grigorescu, S. M., and F. Moldoveanu. 2011. "Controlling Depth Estimation for Robust Robotic Perception." Paper presented at 18th IFAC World Congress, Milano, Italy, 2017. doi:10.3182/20110828-6-IT-1002.00436.

Isermann, R. 2006. *Fault Diagnosis Systems*. Berlin Heidelberg: Springer-Verlag.

Jacobsen, D., and P. Ott. 2017. "Cloud Architecture for Industrial Image Processing: Platform for Real-time Inline Quality Assurance." Paper presented at IEEE 15th International Conference on Industrial Informatics (INDIN), July 24–26, Emden, Germany.

Janabi-Sharifi, F., and M. Marey. 2010. "A Kalman-Filter-Based Method for Pose Estimation in Visual Servoing." *IEEE Transaction on Robotics* 26 (5): 939–947. doi:10.1109/TRO.2010.2061290.

Lee, S. H., and C. S. Yang. 2017. "A Real Time Object Recognition and Counting System for Smart Industrial Camera Sensor." *IEEE Sensors Journal* 17 (8). doi:10.1109/JSEN.2017.2671457.

Li, D. Y., W. C. Cai, Y. D. Song, and H. J. Chen. 2018. "Adaptive Fault-Tolerant Control of Wind Turbines with Guaranteed Transient Performance considering Active Power Control of Wind Farms." *IEEE Transaction on Industrial Electronics* 65 (4): 3275–3285. doi:10.1109/TIE.2017.2748036.

Liu, B., M. Saif, and H. Fan. 2016. "Adaptive Fault Tolerant Control of a Half-Car Active Suspension Systems Subject to Random Actuator Failures." *IEEE ASME Transactions on Mechatronics* 21 (6): 2847–2857. doi:10.1109/TMECH.2016.2587159.

Liu, Y. H., H. Wang, C. Wang, and K. K. Lam. 2006. "Uncalibrated Visual Servoing of Robots Using a Depth-Independent Interaction Matrix." *IEEE Transaction on Robotics* 22 (4): 804–817. doi:10.1109/TRO.2006.878788.

Mahmoud, M., J. Jiang, and Y. Zhang. 2003. *Active Fault Tolerant Control Systems Stochastic Analysis and Synthesis*. Berlin Heidelberg: Springer Verlag.

Murao, T., H. Kawal, and M. Fujita. 2009. "Stabilizing Predictive Visual Feedback Control for Fixed Camera Systems." *Electronics and Communications in Japan* 129-C (4): 1–11. doi:10.1002/ecj.10091.

Noura, H., D. Theilliol, J. C. Ponsart, and A. Chamseddine. 2009. *Fault-tolerant Control Systems Design and Practical Applications*. London: Springer-Verlag.

Prieto, I. G., M. J. Duran, and F. J. Barrero. 2017. "Fault-Tolerant Control of Six-Phase Induction Motor Drives with Variable Current Injection." *IEEE Transactions on Power Electronics* 32 (10): 7894–7903. doi:10.1109/TPEL.2016.2639070.

Qiao, H., Z. He, Z. Zhang, and Y. Zi. 2007. "Fault Diagnosis of Rotating Machinery Based on Improved Wavelet Package Transform and SVMs Ensemble." *Mechanical Systems and Signal Processing* 21 (2): 688–705. doi:10.1016/j.ymssp.2006.01.007.

Schwab, K. 2017. *The Fourth Industrial Revolution*. New York: Crown Business.

Sekar, R., S. J. Hsieh, and Z. Wu. 2011. "Remote Diagnosis Design for a PLC-based Automated System: 1-implementation of Three Levels of Architectures." *International Journal of Advanced Manufacturing Technology* 57 (5): 683–700. doi:10.1007/s00170-011-3314-6.

Shleymovich, M. P., M. V. Medvedev, and S. A. Lyasheva. 2016. "Object Detection in the Images in Industrial Process Control Systems Based on Salient Points of Wavelet Transform Analysis." Paper presented at 2nd International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), May 19–20, Chelyabinsk, the Russian Federation.

Szkilnyk, G., K. Hughes, H. A. Fernando, and B. Surgenor. 2012. "Video Event Detection for Fault Monitoring in Assembly Automation." *International Journal of Intelligent Systems Technologies and Applications* 13 (1/2): 20–25. doi:10.1504/IJISTA.2014.059302.

Tombari, F., L. D. Stefano, S. Mattoccia, and A. Galanti. 2008. "Performance Evaluation of Robust Matching Measures." Paper presented at International Conference on Computer Vision Theory and Applications (VISAPP), Funchal, January 22–25.

Van, M., D. Wu, S. S. Ge, and H. Ren. 2016. "Fault Diagnosis in Image-Based Visual Servoing with Eye-In-Hand Configurations Using Kalman Filter." *IEEE Transaction on Industrial Informatics* 12 (6): 1998–2007. doi:10.1109/TII.2016.2590338.

Wang, C., Q. Liao, and Y. Mei. 2008. "Dynamic Feedback Tracking Control of Nonholonomic Mobile Robots with Unknown Camera Parameters." Paper presented at Proceedings of the 17th World Congress, Seoul, Korea, July 6–11. doi:10.3182/20080706-5-KR-1001.01342.

Wang, H., C. Vasseur, V. Koncar, A. Chamroo, and N. Christov. 2010. "Modelling and Trajectory Tracking Control of A 2-DOF Vision Based Inverted Pendulum." *Control Engineering and Applied Informatics* 12 (3): 59–66.

Wang, T., H. Gao, and J. Qiu. 2016. "A Combined Fault-Tolerant and Predictive Control for Network-Based Industrial Processes." *IEEE Transaction on Industrial Electronics* 63 (4): 2529–2536.

Widodo, A., B. Yang, and T. Han. 2007. "Combination of Independent Component Analysis and Support Vector Machines for Intelligent Faults Diagnosis of Induction Motors." *Expert Systems with Applications* 32 (2): 299–312. doi:10.1016/j.eswa.2005.11.031.

Yu, X., P. Li, and Y. Zhang. 2018. "The Design of Fixed-Time Observer and Finite-Time Fault-Tolerant Controller Hypersonic Gliding Vehicles." *IEEE Transactions on Industrial Electronics* 65 (5): 4135–4144. doi:10.1109/TIE.2017.2772192.

Zhang, L., L. B. Jack, and A. K. Nandi. 2005. "Fault Detection Using Genetic Programming." *Mechanical Systems and Signal Processing* 19 (2): 271–289. doi:10.1016/j.ymssp.2004.03.002.

Zhang, Y., and J. Jiang. 2008. "Bibliographical Review on Reconfigurable Fault-tolerant Control Systems." *Annual Reviews in Control* 32: 292–252. doi:10.1016/j.arcontrol.2008.03.008.