AUTOMATIC DETECTION OF ABANDONED OBJECTS WITH A MOVING CAMERA USING MULTISCALE VIDEO ANALYSIS

Gustavo Henrique Fraga de Carvalho

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Elétrica.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Rio de Janeiro
Junho de 2015

AUTOMATIC DETECTION OF ABANDONED OBJECTS WITH A MOVING CAMERA USING MULTISCALE VIDEO ANALYSIS

Gustavo Henrique Fraga de Carvalho

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Eduardo Antônio Barros da Silva, Ph.D.

Prof. Sergio Lima Netto, Ph.D.

Prof. Hélio Côrtes Vieira Lopes, D.Sc.

Prof. José Gabriel Rodriguez Carneiro Gomes, Ph.D.

Prof. Siome Klein Goldenstein, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
JUNHO DE 2015

Carvalho, Gustavo Henrique Fraga de

Automatic Detection of Abandoned Objects with a Moving Camera Using Multiscale Video Analysis/Gustavo Henrique Fraga de Carvalho. – Rio de Janeiro: UFRJ/COPPE, 2015.

XV, 98 p.: il.; 29, 7cm.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2015.

Bibliography: p. 92 – 98.

1. video surveillance. 2. moving camera. 3. abandoned object detection. 4. cluttered environment. I. Silva, Eduardo Antônio Barros da *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Acknowledgements

I would like to thank my advisors, Professor Eduardo Antônio Barros da Silva and Professor Sergio Lima Netto, for their patience, support, knowledge and guidance.

Besides my advisors, I am thankful to my colleagues, José Fernando de Oliveira, Allan Freitas da Silva, Lucas Thomaz, Eric Jardim and Mateus Nakahata, without whom the completion of this thesis would have been extremely more difficult.

Also, I thank my other colleagues from the Signal Multimedia and Telecommunications Laboratory, who helped me whenever possible.

My gratitude also goes to the students, researches and professors from the GSCAR (Grupo de Simulação e Controle em Automação e Robótica), who set up the robotic platform used for the production of the database employed in this thesis.

I am also grateful to my family for their endless support.

Finally, I would like to thank everyone who directly or indirectly helped me during this enterprise.

Once again, thank you all.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

DETECÇÃO AUTOMÁTICA DE OBJETOS ABANDONADOS COM CÂMERA EM MOVIMENTO USANDO ANÁLISE DE VÍDEO MULTIESCALA

Gustavo Henrique Fraga de Carvalho

Junho/2015

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Programa: Engenharia Elétrica

Esta tese aborda o problema da detecção de objetos abandonados em um ambiente desordenado usando uma câmera montada sobre uma plataforma robótica que executa um movimento de translação de ida e volta ao longo de um trilho retilíneo. Neste cenário, o sistema desenvolvido compara as imagens captadas com um vídeo de referência, exigindo, portanto, adequados alinhamento temporal e registro geométrico entre os dois sinais. Uma restrição de tempo real é imposta ao sistema para permitir que este seja capaz de executar tarefas de vigilância eficazmente em situações práticas. Neste estudo, nos propomos a lidar com a detecção simultânea de objetos de diferentes tamanhos usando uma abordagem multirresolução, juntamente com correlação cruzada normalizada e uma etapa de votação. A fim de desenvolver e avaliar adequadamente o método proposto nós projetamos, gravamos e marcamos uma base de dados produzida em um cenário de vigilância real, que consiste em um galpão industrial contendo um grande número de tubos e máquinas rotativas. Além disso, desenvolvemos uma rotina de ajuste de parâmetros sistemática que permite que o sistema seja adaptado para diferentes cenários. Nós a validamos empregando a base de dados produzida. Os resultados obtidos são bastante efetivos, apresentando detecção de objetos abandonados robusta no ambiente industrial proposto.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

AUTOMATIC DETECTION OF ABANDONED OBJECTS WITH A MOVING CAMERA USING MULTISCALE VIDEO ANALYSIS

Gustavo Henrique Fraga de Carvalho

June/2015

Advisors: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Department: Electrical Engineering

This thesis addresses the problem of abandoned object detection in a cluttered environment using a camera mounted on a robot platform that performs a translational back-and-forth movement along a horizontal straight track. In this scenario, the developed system compares captured images to a reference video, thus requiring proper temporal alignment and geometric registration between the two signals. A real-time constraint is imposed onto the system to allow an effective surveillance capability in practical situations. In this study, we propose to deal with the simultaneous detection of objects of different sizes using a multiresolution approach together with normalized cross-correlation and a voting step. In order to develop and properly assess the proposed method we designed, recorded and annotated a database produced in a real surveillance scenario, consisting of an industrial plant containing a large number of pipes and rotating machines. Also, we have devised a systematic parameter tuning routine that allows the system to be adapted to different scenarios. We have validated it using the designed database. The obtained results are quite effective, achieving robust abandoned object detection in the proposed industrial plant scenario.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, technology can provide us with large amounts of different kinds of data, that we can employ in many different ways.

Considering only the image domain, for example, we have still images, videos obtained from static cameras, videos obtained from moving cameras, images or videos obtained with multiple cameras, multidimensional data obtained from medical scanners, and so on.

In this context, Computer Vision techniques allow one to extract reliable information from a video signal, providing a reasonably good understanding of the recorded environments or processes. By exploring this capability, automatic video-based monitoring systems can be deployed, enabling significant saving of resources, minimizing labor risks in hazardous environments and increasing system efficiency, particularly when dealing with repetitive or tedious tasks.

In this monitoring application, the automatic detection of abandoned objects in a given scenario constitutes an interesting feature of a surveillance or remote inspection system. When using static cameras for this purpose, for instance, simple background and behavior subtractions, employing statistical approaches, allow one to detect or even track objects in the acquired videos [1–6]. Using moving cameras in this context, however, increases the surveillance range [7], but requires more elaborate techniques to compensate for the camera movement, which may also include a significant amount of vibration [8–13]. In addition, if the environments to be monitored are cluttered, the process of sorting out the useful information from the background becomes even more difficult, reducing the overall detection robustness.

1.1 Proposed Scenario

An example of cluttered environment that could greatly benefit from the use of Computer Vision techniques are industrial plants. In such hazardous places, remote

inspection tasks, for example, would not only improve their economic operation, but would also help minimize labor risks, as stated before.

Concerning the issue of increasing the surveillance range, we envisioned that moving cameras would be an even greater help to the performance of remote inspection tasks in such places. Also, considering the fact that there are people performing inspection tasks in shifts in such environments, we believe that a camera being moved by a robot, performing the path of the inspector around the environment could not only reduce his workload, but also allow him to perform this task from a safe place, reducing the need of his presence in the hazardous environment.

Regarding the cost issue, industrial plants such as oil platforms could use cameras capable of detecting hydrocarbon gases in order to try to find gas leakage. As this kind of camera can cost 200 thousand dollars, increasing the surveillance range of such a camera, attaching it to a moving platform, for example, could lead to the saving of resources.

Considering the automatic detection of abandoned objects concept presented in this thesis, with further research, it could be extended to the automatic detection of other kinds of anomalies, such as fire, smoke, liquid or gas leakage (as in the example above), etc. Such anomalies, being present in a certain number of frames, at roughly the same position in the monitored environment, could be regarded as an abandoned object at that same position. The automatic detection of such anomalies would be extremely helpful, not only because their fast detection would allow a faster solution to the problem, helping save resources and preventing economic losses, even diminishing the harmful impact to the environment in some cases, but also because it could help prevent that people would be subject to dangerous situations.

With this industrial plant scenario in mind, we proposed a surveillance system and produced a database recorded in a real-world industrial (cluttered) environment, to help us develop and test the system.

1.2 Surveillance System

In this work, we describe a complete surveillance system for detecting the presence of abandoned objects in a cluttered environment. Our system employs a camera attached to a robotic platform mounted over a horizontal straight track performing a translational back-and-forth movement. The monitoring system uses a reference video with no unexpected object, as certified by a system operator in an initial calibration stage. The detection of anomalous objects is carried out by comparing the target video, acquired in subsequent performances of the robotic platform trajectory, with the initial reference video. As our system must perform a monitoring task in an environment such as an industrial plant, it must be able to send alarms as a

human inspector performing the same task would. Throughout this thesis, this will be our definition of real time. Our system, then, cannot take more than a couple of minutes to send an alarm if an abandoned object is detected in the region being monitored at the time. Such real-time constraint requires quite specific and demanding signal-processing solutions and makes the system suitable for a wide scope of practical situations. This way, this desirable feature should be attainable after the optimization of the programing code developed during the research presented in this thesis.

During the development of our object-detecting method, we employed a database (described in Chapter 4) composed by reference and target videos that presented only small degrees of changes in illumination among them. Some of these differences were caused by the the changes that occur during different times of the day (morning and afternoon), while others were generated by the use of a spotlight. Thus, greater variations in illumination are not a part of our premises.

A relevant issue in such surveillance systems is the temporal alignment of the reference and target videos. Solutions to this problem usually include external trigger signals to determine the camera position, such as a GPS device [8] or the robot's odometry [14, 15]. This work dispenses with external signals for temporal alignment, the camera position being determined using a maximum-likelihood model for the camera movement derived directly from the acquired reference and target videos.

A multiscale approach is proposed to compare the synchronized and registered frames from the reference and target videos. In this framework, larger abandoned objects are searched in lower video resolutions and smaller objects are searched in higher resolutions, leading to an increased detection robustness at a reasonable computational cost.

Video comparison includes the computation of the normalized cross-correlation (NCC) [8] between two video frames within the proposed multiresolution approach. After an NCC threshold operation, a binary detection mask is determined. Subsequent nonlinear operations, which include a temporal filtering, voting step, and morphological operations, remove most false positive and false negative situations, increasing the overall system accuracy.

A complete strategy is presented for setting the system parameters in order to maximize its detection rate. In such description, the impact of each system variable on the resulting performance is thoroughly evaluated. System performance is assessed using validation on part of a large database, recorded in a real industrial plant, comprising more than 8 hours of annotated video and several types (different colors, sizes, positions etc.) of abandoned objects, as detailed in [16].

To describe the proposed surveillance techniques, this document is organized as follows: Chapter 2 briefly addresses related problems and their proposed solutions.

Chapter 3 presents the deployed system, including the video-comparison strategy in a step-by-step procedure. Chapter 4 briefly describes the database employed to adjust and evaluate the proposed detection scheme. Chapter 5 details all specific solutions developed in the context of this work to optimize the system's performance in terms of computational complexity and detection robustness. Chapter 6 describes the configuration of all system variables of interest, discussing their individual effects on the resulting detection process. In Chapter 7, detection results are presented characterizing the system's performance in both quantitative and qualitative ways. Finally, Chapter 8 concludes this work emphasizing its main technical contributions.

Chapter 2

Studied Techniques

In this chapter we will present some of the techniques we studied having in mind our application of remote monitoring of industrial plants.

2.1 Static Cameras

The easiest way to monitor a desired environment is to employ a static camera. To perform automatic surveillance tasks with images obtained from such cameras, one can use, for example, environment modeling and motion segmentation [17–20].

Motion segmentation, that is, the separation of a moving object in an image from a static background, can be performed through three main approaches: temporal differentiation [21], background subtraction [18, 19, 22–24], and optical flow [17]. Temporal differentiation uses the pixel difference between 2 to 3 consecutive frames to extract the motion regions [25]. This method is fast and adaptive to dynamic environments, but often cannot extract all the relevant pixels [17, 22].

In background subtraction, the pixel to pixel difference between the current image and the reference image is calculated. This method is very sensitive to changes, such as variations in illumination. Statistical methods present robustness to noise, shadows, lighting changes, etc [17, 22].

Optical flow is the pattern of apparent motion of objects, surfaces and edges in a scene caused by the relative motion between an observer and the scene. It can be calculated based on gradients, phase correlation, block-based or differential-based methods. In general, the use of optical flow requires special hardware for real-time applications. It is computationally expensive and sensitive to noise [17, 22].

In the case of motion segmentation, specifically, several algorithms are proposed for its implementation, employing, for example, adaptive mixed Gaussian models [17, 20, 22, 24–30], in which each pixel of the background is modeled as a weighted mixture of Gaussian distributions. When a new image is acquired, its pixels are compared to their corresponding models. Based on the mean and variance of each

Gaussian in its model, it is decided if the new pixel belongs to the background or not.

Other techniques to statistically model pixels employ Bayesian modeling; Kalman filters [31]; statistical models representing pixels with three values (maximum, minimum, and maximum difference in intensity between the pixel and its corresponding one in the previous frame); models that classify pixels as belonging to the background, the foreground, or as shadows [19, 24, 32, 33]; or combinations of the models presented.

If one considers the most complex problems to be addressed, such as analysis and understanding of behavior and activities, further different techniques are needed. Behaviour subtraction [5] intends to deal with issues caused by small variations in the background, produced by jitter, small camera trepidations, and so on. Here, anomalous events are detected by subtracting the behaviour model of a scene, produced by capturing the scene dynamics at specific areas in the images within determined time windows, from the one of a given frame of interest.

Other techniques to deal with analysing and understanding behaviour and activities include neural networks, hidden Markov models [30], vector quantization, Markov random fields, dynamic time warping, support vector machines [30], among others [34].

As they were developed considering static cameras, all the techniques presented above deal with stationary backgrounds. If one intends to monitor a large area, and it is not possible to employ multiple static cameras to cover it (due to cost restrictions, for example), moving cameras must be used.

2.2 Moving Cameras

With moving cameras, the image background is not stationary. This way, different techniques must be used to deal with this issue.

2.2.1 PTZ Cameras

Pan-Tilt-Zoom (PTZ) cameras generally allow a pan of 360°, a tilt of 90°, and some optical zoom.

In [10], background compensation with a PTZ camera is studied. It approximates the relationship between consecutive images through similarity transformations. The outlier removal is done using Hough transformation together with RANSAC [35].

In [36], a technique for Motion Tracking with PTZ cameras is described. It employs image mapping to align the images and allow the use of techniques employed with static cameras. Morphological filtering is used to desensitize the detection

algorithm to small issues with the background compensation. The method presented needs the angle information (pan and tilt) in order to work, and the camera can move a maximum of 3 degrees between each frame.

A method to track objects and also estimate the camera movement is described in [37]. In this study, the camera can be either a PTZ one, or a camera that moves along a path. The studied topics are formulated as a Bayesian motion-based partitioning problem in the spatiotemporal domain of the image sequence. The coupling of the object tracking and motion estimation problems is explored through a greedy method for movement parameters estimation.

In [38], real-time object detection and tracking is performed with a moving camera. Spherical coordinates are centered in the camera, and the original image coordinate is transformed to a spherical coordinate. No background subtraction technique is employed. The multiple target tracking with the PTZ camera is formalized in a probabilistic fashion. Hidden Markov Models (HMM) are used to represent the dependencies between the targets and the camera at the spherical coordinates (the transition between states is modeled as a Markov process). Recursive Bayesian filtering is also used.

Considering the tracking issue, sequential importance sampling (SIS) and sampling importance re-sampling (SIR) particle filters are used in order to deal with the interactions between near targets. Considering the detection issue, the Kanade-Lucas-Tomasi (KLT) feature tracker is used to obtain the optical flow in the image sequences. This, together with HMM, is employed to obtain moving feature points with a non zero optical flow at the spherical coordinates. Color histogram and contour matching is used in order to finish the detection process. The camera movement is measured from its motor encoder.

2.2.2 Cameras Mounted on Moving Platforms

Besides PTZ cameras, one can also monitor larger areas with a single camera by mounting it over moving platforms.

In [39], a camera mounted on top of a mobile robot is used in order to detect moving objects and estimate their movements. In this study, background subtraction is performed employing Gaussian mixture models (GMM). Dense motion analysis is used to estimate and compensate the camera movement. Maximum a posteriori probability (MAP) framework is used to detect the moving objects and to estimate their movement field.

In [40], once again, a camera mounted on a moving robot is employed to perform moving object detection. Here, a recursive Bayes filter is used to detect the movement (a first-order Markov process is assumed). The robot's odometry is employed

for image alignment. It must be correctly synchronized with its corresponding frame. Kanade-Lucas-Tomasi (KLT) is used to track features in an image pair. The Bayes filter determines if these features are stationary or dynamic. The object movement considered is degenerate, that is, it must be parallel to the camera movement in order for the proposed method to work. Only grayscale information is used in this method.

In [14], the study initiated in [40] is continued. Visual Simultaneous Localization and Mapping (SLAM) is employed to estimate the camera movement. Once again, a recursive Bayes filter is used to calculate if the feature probabilities that will be used in the SLAM model are static or dynamic. The SLAM framework and RANSAC are employed to estimate the initial epipolar geometry. The method was able to detect a maximum of three objects moving independently in each frame.

In [41], real-time object tracking is performed using a hybrid approach between movement segmentation and searching for known features. A single estimation step is used to filter the background global movement before applying a background filtering algorithm to perform the detection of a moving object. As real-time performance is a goal in this study, the background filtering algorithm must be robust to inadequacies in the motion estimation and compensation steps. After the said motion estimation and compensation steps, object and background segmentation is performed. Then, adaptive noise filtering is employed, and finally the moving object is detected. To track the detected object, the Condensation algorithm is used. It provides a statistical framework for projecting forward the parameters of any model and adjusting the method's previsions based in the actual content of each new frame.

A monochromatic camera attached to a car is employed in [42]. It is assumed that no rotations will occur. The proposed method consists in detecting Points of Interest, calculating the optical flow with the Lucas-Kanade algorithm, robustly estimating the focus of expansion (FOE), computing the residual FOE map, and segmenting it, in order to detect the object movement.

The study described in [43] deals with the detection of moving targets. It uses SIFT [44–46] to extract Points of Interest, and RANSAC to estimate affine transformations. Background subtraction between the actual frame and a transformed one is employed to perform the object detection. The detection result is used to update the background.

The problem described in [8] presents a similar idea: a camera attached to a car is used in order to find abandoned objects at roads. The car moves at 30 km/h. Initially, a predetermined path is recorded, with no abandoned object present in the video (the reference video). Then, new videos are generated (the target videos), recorded in the same path, now with abandoned objects in them. A technique is employed in order to identify the horizon line, as abandoned objects will be searched

only below this line.

The initial video alignment is done via GPS. After that, the frames in the reference and target videos are registered, with the aid of SIFT to extract the Points of Interest, and RANSAC do select them, in order to calculate the homographies between the frames. After this step, the registered frames are compared through the computation of normalized cross correlation (NCC), producing an NCC image that is binarized with an experimental threshold. The results produced here presents possible abandoned object candidates.

To help reduce the occurrence of false positives, a method for removing false alarms on high objects is performed. It is quite similar to the one previously described. However, this time, the registered and compared frames are all from the reference video.

To remove false alarms on the dominant plane, once again, a similar process is used. This time, once again the frames from the reference video are registered and aligned among themselves, but now the same process also happens with the frames from the target video. This time, however, instead of using NCC, the registered frames of both video sequences are compared by calculating the difference of grayscale pixel values (in this step, once again, the reference frames are compared with reference frames, and now, also, target frames are compared with target frames). Finally, after all these techniques are performed, a final temporal filtering step is done to confirm the existence of a suspicious abandoned object in the target video frame sequence.

Considering our scope of automatic abandoned object detection in a cluttered environment, some works [47–49] have been done employing our database [16].

In [47], a system is proposed in order to estimate the trajectory of a moving camera employing a structure from motion algorithm adapted to use images obtained from a camera performing a horizontal linear movement. With the obtained trajectory, a panoramic image is produced. The idea is to compare panoramic images obtained in the same way from the same environment in order to try to find abandoned objects in it.

Based on the concepts presented in [50] and [51], in [48] a framework is proposed in order to detect which frames from a given video might contain abandoned objects in them. Optimized operators that produce Gaussian outputs when the reference video is applied to them are generated from it. When the target video is applied to them, if there is an anomaly present in a given frame sequence, a non-Gaussian output is produced. This method is robust to rotations and translations between the frames and does not require synchronization between the reference and target frames.

In [49], Robust Subspace Recovery [52] is used to produce a low-rank representa-

tion of a reference video signal. Using it to represent a target video signal containing abandoned objects in some of its frames, an error signal, representing the abandoned objects and some high-frequency data, is produced. As this high-frequency data is common between the reference and target videos, this error signal can be decomposed, and the abandoned object information can be retrieved from it. This method also does not require synchronization between the reference and target frames.

None of the techniques outlined in this chapter address the problem of detecting abandoned objects in a cluttered environment (specifically, an industrial plant) in real time using a moving camera, as exposed in Section 1.2. This way, Chapter 3 will present the framework proposed in this thesis in order to deal with the outlined problem. Also, Chapter 4 will present the database that had to be planned, recorded and annotated so that we could properly develop our algorithms.

Chapter 3

General System Description

The proposed surveillance system consists of a high-definition (HD) 24 frame/second camera mounted on a robotic Roomba® platform [53] performing a back-and-forth movement on a horizontal track, as illustrated in Fig. 3.1. The moving robot takes about 3 minutes to cover the entire 6 m track, which oversees an industrial plant with a cluttered background.

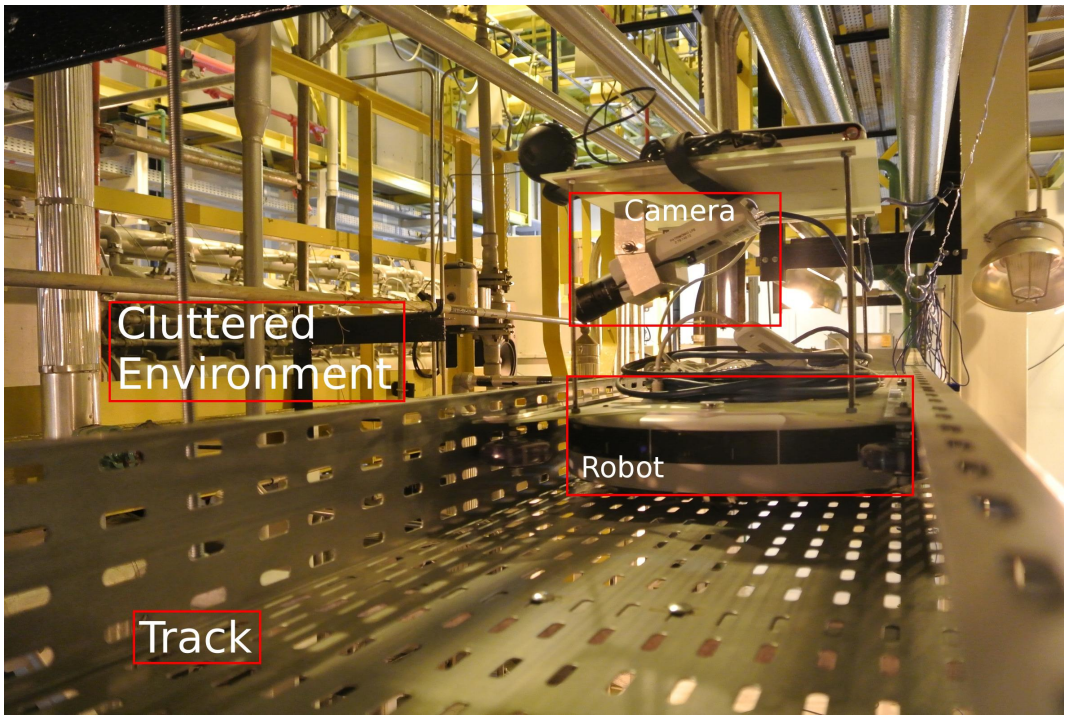


Figure 3.1: Real camera-robot system with a glimpse of the cluttered environment of interest on the background.

The following framework was employed for the real-time system operation: a reference video is obtained from an initial performance of the robot back-and-forth trajectory and is validated by some operator, indicating the absence of any strange object. The videos from all subsequent performances of the robot trajectory are

then compared to that reference video in search of any newly observed object. If necessary, the system operator may change the reference video using a simple update procedure, and after this the monitoring system goes back to normal operation.

For a proper object detection within a given video sequence, the developed system includes the following processing steps:

(i) Reference and target video synchronization, both for the initial alignment and for correcting subsequent deviations due to small variations on the robot's speed;

(ii) Identification of points of interest (PoI), usually salient points, in the corresponding reference and target frames to allow simplified real-time processing;

(iii) Geometric registration between the corresponding reference and target frames to reduce vibration effects on the robot movement along the cable trail;

(iv) Numerically efficient (for real-time purposes) and robust frame comparison, using a multiresolution scheme, to identify significant frame discrepancies which can be associated to an abandoned object;

(v) A pixel-level voting strategy along consecutive detection masks to identify consistent detections along time, removing occasional false detections;

(vi) Morphological operations to remove additional false positive and false negative situations.

Details about the techniques proposed in this study for implementing the above stages are presented in Chapter 5.

Fig. 3.2 shows a block diagram representing our abandoned object detection process.

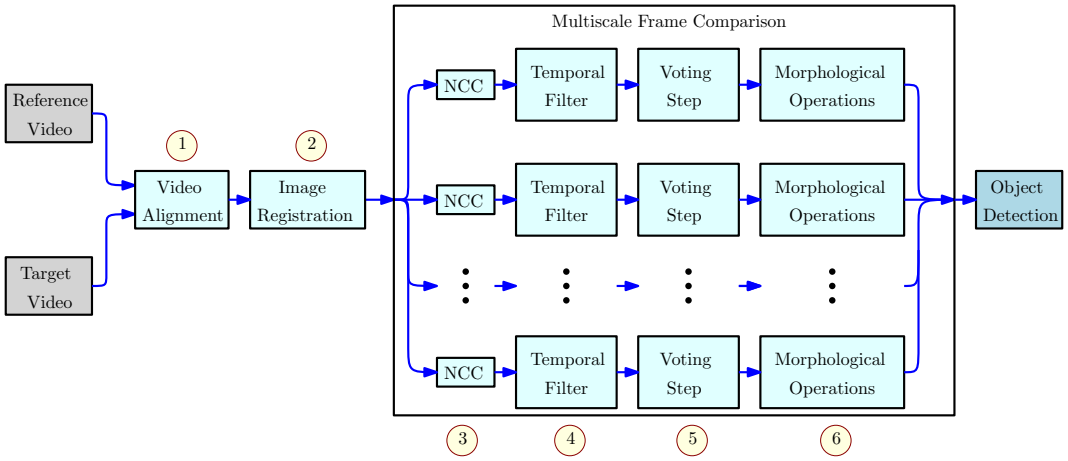


Figure 3.2: Block diagram of the system abandoned object detection process.

Block n.1 represents the reference and target video synchronization. Block n.2 represents the PoI identification process, and the geometric registration between the corresponding reference and target frames. In our multiresolution approach, block n.3 shows our frame comparison step, employing NCC, block n.4 our temporal filtering step, block n.5 our pixel-level voting strategy, and finally in block n.6 the morphological operations are performed. Together, the results obtained in all resolutions indicate where abandoned objects might be.

Fig. 3.3 shows another view of our real camera-robot system setup in the cluttered environment where the database described in Chapter 4 was recorded.

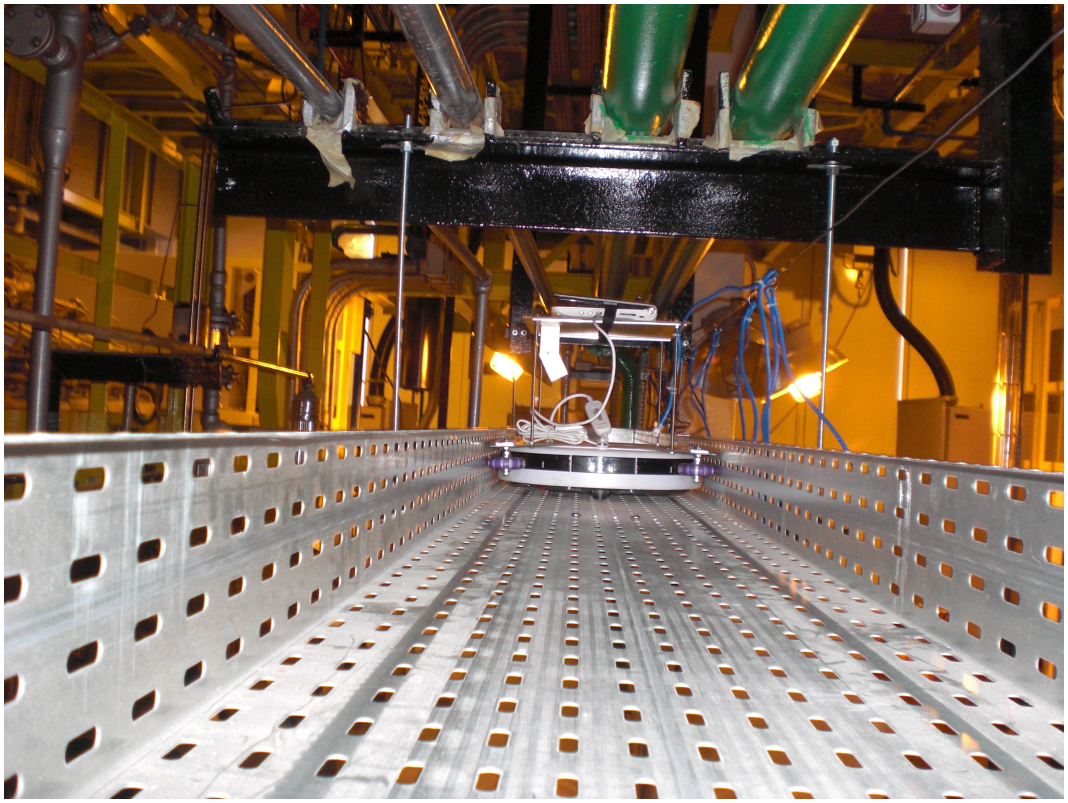


Figure 3.3: Another view of our real camera-robot system setup.

Chapter 4

Abandoned-Object Video Database

In order to allow a systematic verification of the system robustness, a large video database, described in [16] and available at [54], was deployed. We refer to it as the VDAO database (from "video database of abandoned objects in a cluttered industrial environment"). The whole database comprises more than 8 hours of video, including 8 reference videos (without abandoned objects) and 57 different target videos (3 multi-object videos and 54 single object videos).

4.1 Database Development

4.1.1 Database Requirements

We developed this database with the following requirements in mind:

- Videos should include several changes in the movement direction to allow a proper assessment of the time-alignment algorithm;
- Objects of different colors, textures, shapes and sizes must be considered;
- The positions and numbers of objects within a frame must change in different video footages;
- Small differences in the levels of luminosity must be considered, not only in different videos but also within a same video footage.

4.1.2 Database Design

Based on the requirements listed in Subsection 4.1.1, the proposed database was devised in the following way:

- Use of two different cameras, both set to a resolution of 1280 x 720 pixels and a rate of 24 frames per second, with quite distinct light/color characteristics;

- Use of a spotlight in half of the footages, making brighter recordings, whereas natural light was employed on the other half of the recordings;
- Recording of 3 multiple-object and 2 no-object (reference) videos with one camera (Dlink DCS-3717) and 54 single-object and 6 no-object (reference) videos with the other camera (Axis P1346);
- All the multiple-object recordings included 6 full performances of the robot trajectory (or 5 direction changes), where the single-object videos only have one trajectory performance in each direction;
- Use of 15 distinct objects for the multiple-object videos (see Fig. 4.1), and 9 other objects in the single-object videos (see Fig. 4.2). Some objects are transparent or reflective, and some are made of a material similar to the environment, so that there is a significant probability that they are not detected by the human eye;
- The 3 multiple-object videos were recorded with the same 15 objects placed in 3 different positions, with the spotlight, making completely distinct arrangements. The different positions also caused some objects to change size in the footages, as they may be farther or closer to the camera. The 54 single-object videos work in a similar way, as each of the 9 objects was recorded in 3 different positions and with/without the use of a spotlight;
- All multiple-object videos were devised in a way that most of the frames include at least 2 objects.

Figure 4.1: Objects used in the multiple-object videos (scales have been changed for a better presentation): (a) string roll; (b) bag; (c) white box; (d) lamp-bulb box; (e) spotlight box; (f) mug; (g) blue coat; (h) wrench; (i) bottle; (j) blue box; (k) backpack; (l) pink backpack; (m) bottle cap; (n) umbrella; (o) green box.

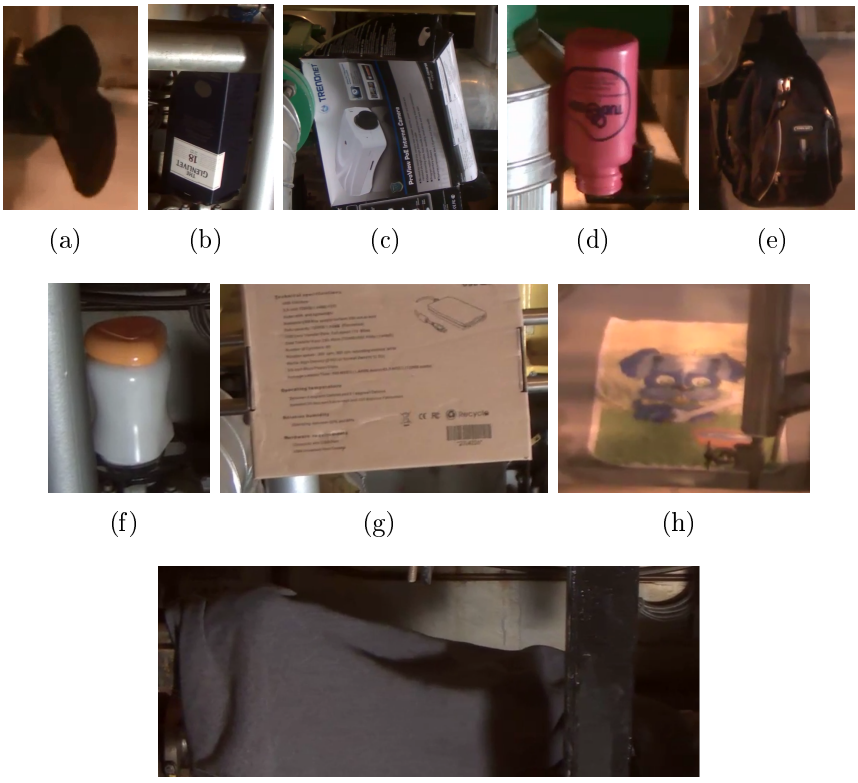


Figure 4.2: Objects used in the single-object videos (scales have been changed for a better presentation): (a) shoe; (b) dark blue box; (c) camera box; (d) pink bottle; (e) black backpack; (f) white jar; (g) brown box; (h) towel; (i) black coat.

4.1.3 Database Recording

The database recording was performed with the camera mounted on an iRobot Roomba® platform passing over a hanging rail at a height of approximately 2.5 m, as seen in Chapter 3. The camera was pointed to a cluttered environment comprising several pipes and valves simulating a scene of interest inside an offshore facility, as shown in Figs. 4.3 and 4.4.

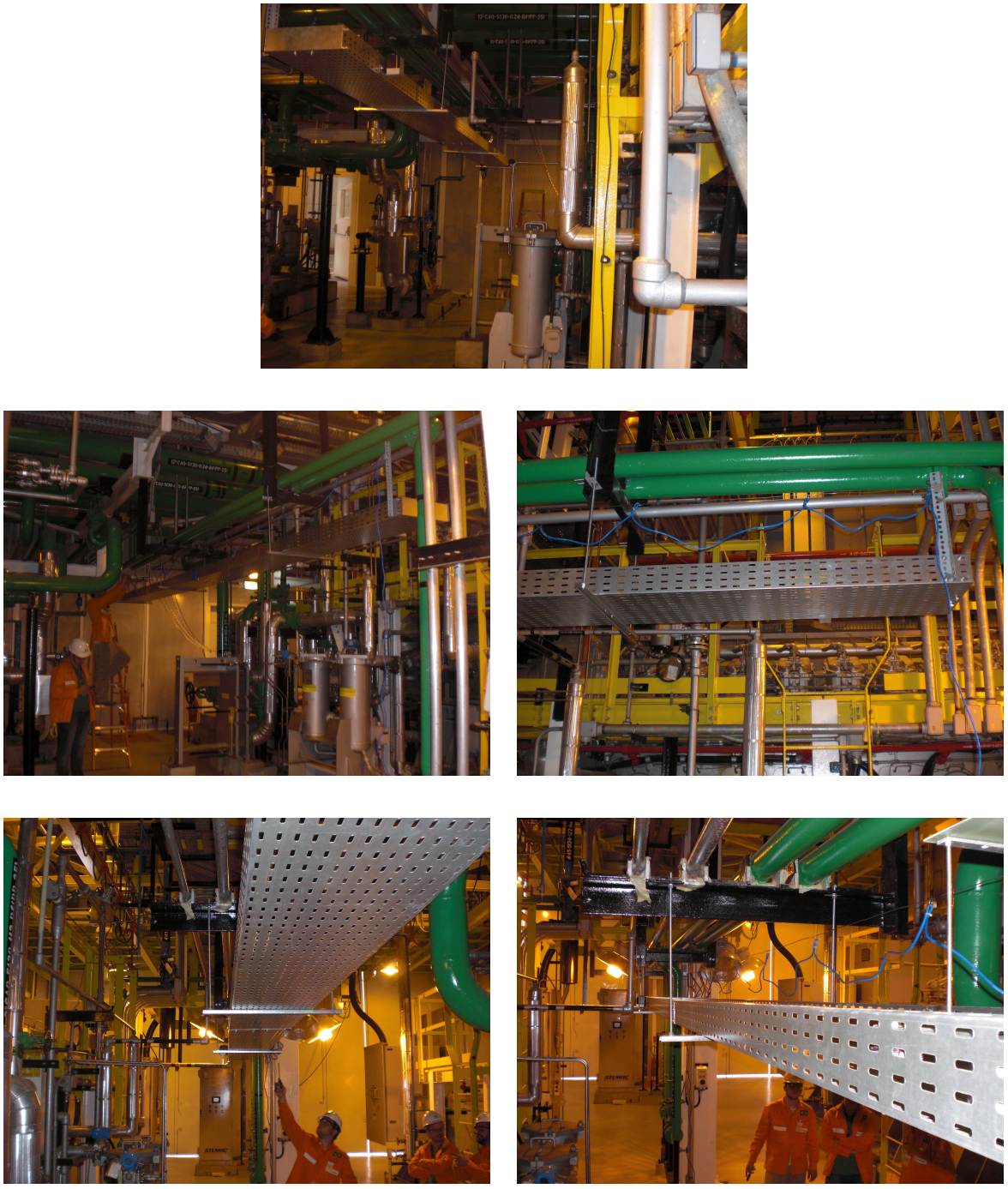


Figure 4.3: Environment where our database was generated.

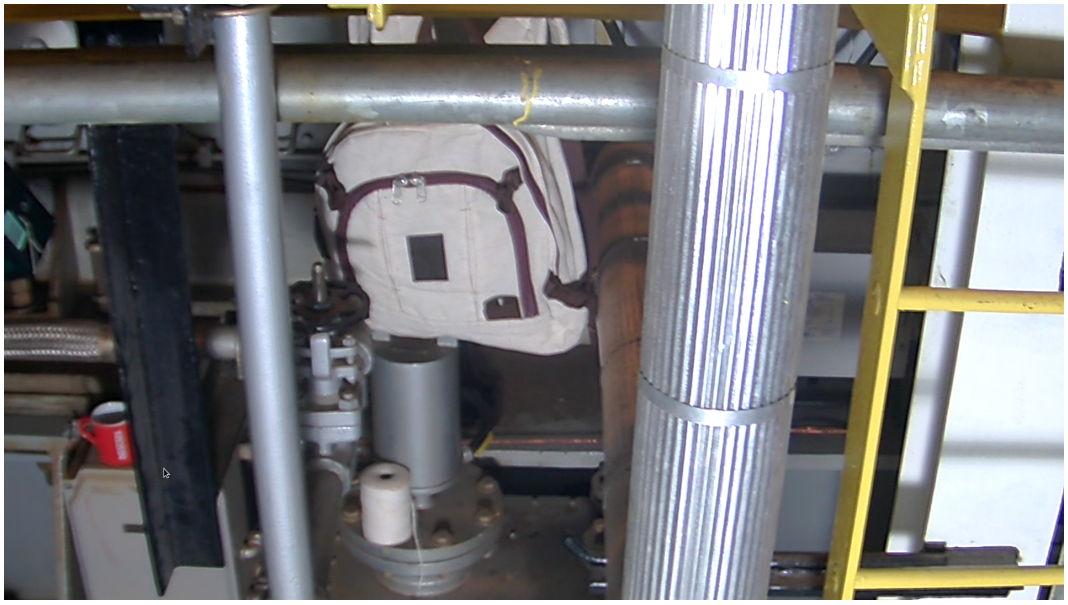
Figure 4.4: Environment where our database was generated.

Slight differences in illumination, video durations and robot speeds can be identified within the database, as the recordings were performed in several sessions comprising a total period of about 7 months. These recordings were performed in different times of the day, which also contributed to the differences in illumination, as daylight varies from morning to afternoon, even inside the relatively closed industrial plant environment where the recordings took place. Such differences are interesting in the sense that they allow a more robust evaluation of the proposed system with respect to the intended characteristics.

It is also important to mention that the Roomba® robot was built having in mind that it would only move in the forward direction, that is, its motor would not, normally, rotate in the opposite direction. This way, while moving backwards, the robot's motor does not operate as smoothly as when it is moving in the forward direction.

Due to the cluttered characteristic of the environment, many object occlusions occur during the videos. An example is when part of the scenario (or of a foreground object) passes in front of (another) object of interest, as seen in Figs. 4.5 and 4.6. This challenge is also a desired characteristic of the database, since the surveillance system should be able to detect an abandoned object even if it is occluded during a

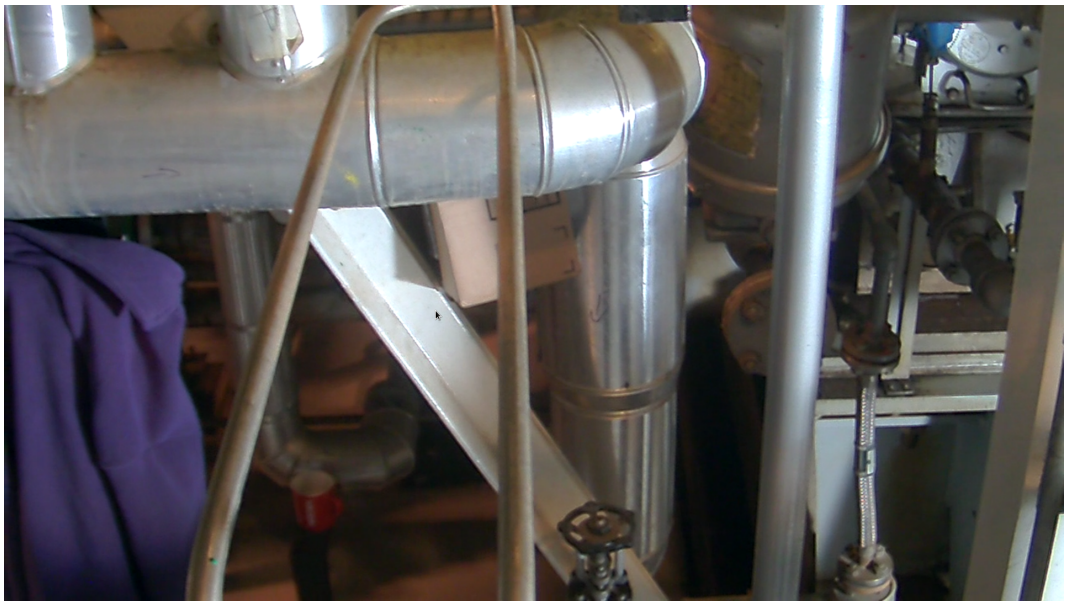
small period of time.



(a)

(b)

Figure 4.5: Example of occlusion: (a) no occlusion is happening; (b) foreground object (pipe) is occluding the string roll and partially occluding the backpack.



(a)

(b)

Figure 4.6: Another example of occlusion: (a) spotlight box is being partially occluded by pipe; (b) blue coat (abandoned object) is occluding the mug.

One (out of the 3) multiple-object videos has only 5 camera trajectory performances instead of 6. The single-object videos have an average duration of 6 minutes, where the multiple-object videos have an average duration of 18 minutes (except the one with 5 trajectory performances that is about 15 minutes long).

Another interesting feature of the database is that the recorded videos include unwanted images of the objects, due to the many reflective surfaces of the cluttered environment, as seen in Fig. 4.7, and shadows casted by the objects or the scenario itself, as seen in Fig. 4.8. These effects may impair the performance of the surveillance system, by causing false positive or false negative situations.

Figure 4.7: Example of reflective surface (pipe on the right, immediately to the right of the green pipe). The backpack to its left is being reflected on it.

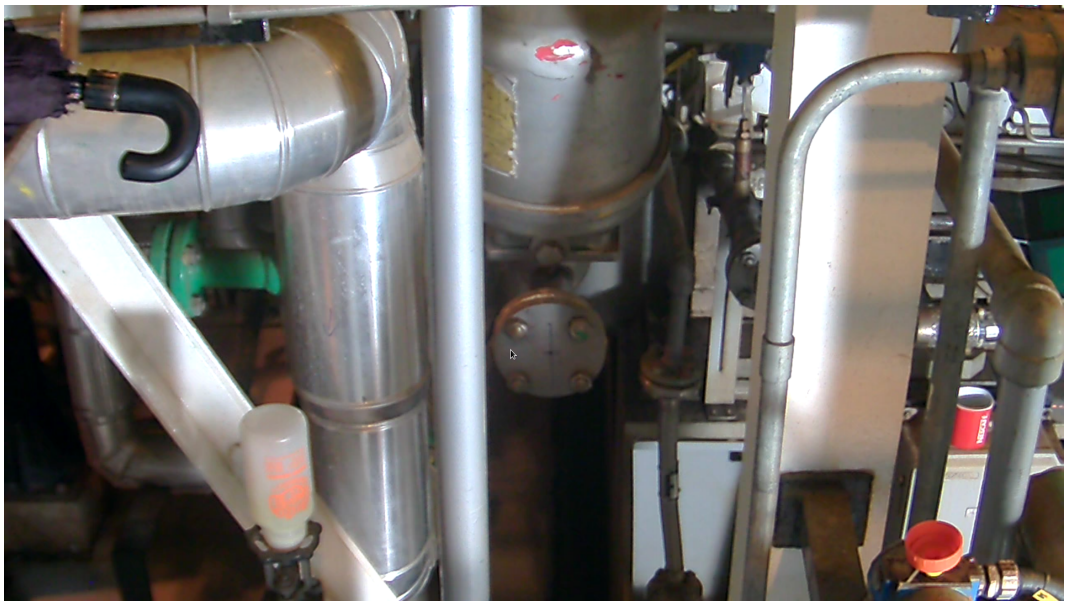


Figure 4.8: Example of shadow being cast by an object, the bottle.

4.2 Database Annotation

A crucial step to turn the proposed database into a useful tool for the performance assessment of object-detecting systems is to identify all objects within its video footages. In our case, we opted for a manual annotation process of each object, since the human ability is considered the gold standard for the application at hand. Due to the large quantity of frames to be marked (in the order of 7.1×10^5 frames), it became clear that the support of a specific software for this purpose, with the following characteristics, would be necessary:

- Object mark consisting of an outline of easy identification, preferably a simple bounding box;
- Marks inserted quickly, via mouse, due to the large number of frames to be considered;
- Input commands via a GUI interface with a minimum number of intuitive commands;
- Ability to mark multiple objects in the same frame;
- Possibility of identifying and associating several parts to a single object due to occasional partial occlusions;
- Generation of an output file with the labels and corresponding coordinates of all objects in each frame.

A new marking software attending all these requirements was developed in C++ [55], using the free version of the QT Creator 2.81 application [56], due to its friendly interface, extensive documentation available and portability between Windows and Linux (Fedora 19) operating systems. The program also used the free and multiplatform OpenCV library [57] for video manipulations. The main screen of the developed annotation tool is shown in Fig. 4.9, whose main commands are indicated on the left side (from top to bottom):

1. Set video position (in frames) by a slider (gross adjustment) or frame number (fine adjustment);
2. Open video file to be marked;
3. Play video (with marked objects if a corresponding markup file exists);
4. Save the markup text file;
5. Set the object name and sub-index;
6. Skip frames without manual mark (interpolation is performed, as described below);
7. Clear specific object from the output file;
8. Set video frame rate.

The marking process of a given video starts by opening the desired file and setting it to the desired frame position. When first marking a video, the frame position should be set to 1; a different frame number is used when marking a video which has already been partially marked. The play-video function can be used to determine the video portion which has already been marked.

For marking an object, a rectangular bounding box was chosen for its simplicity. For that purpose, the mouse must be positioned at any bounding-box corner and dragged to the opposite corner with the left button pressed. When the left mouse button is released a box is drawn around the object, which is then validated with the "set" button.

The name of the object and its sub-index should be informed by the user. In the proposed syntax, full objects are marked with sub-index zero. In case an object is obstructed by another and it seems to be divided into several parts, each part receives a sub-index number varying from 1 to the number of parts.

To expedite the marking process, the user can skip a given number of frames and have the annotation tool to generate bounding boxes at interpolated positions (supposing they are at a constant speed). Although such interpolation process can

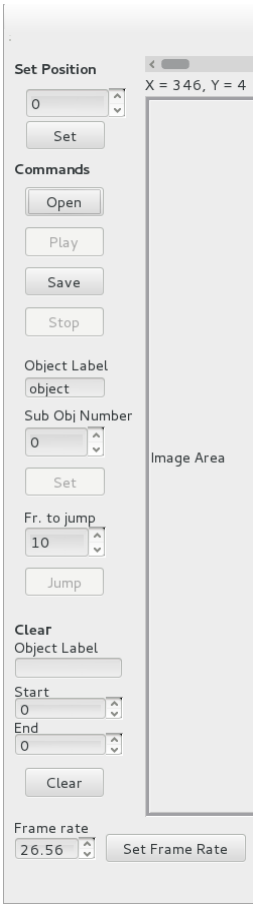


Figure 4.9: Main menu of video annotation tool showing all marking features.

lead to some marking error, at high frame rates (e.g. 24 frames/s), the variation between close frames (about 10 frames apart) is quite small and so is the inserted error.

The entire annotation process is summarized in a text file containing, in each line, the label of the object, the frame number and the coordinates of the upper-left and bottom-right corners of the bounding box for each object or sub-object, as shown in Fig.4.10.

Fig. 4.11 shows an example of annotated frame whithin the developed annotation tool interface.

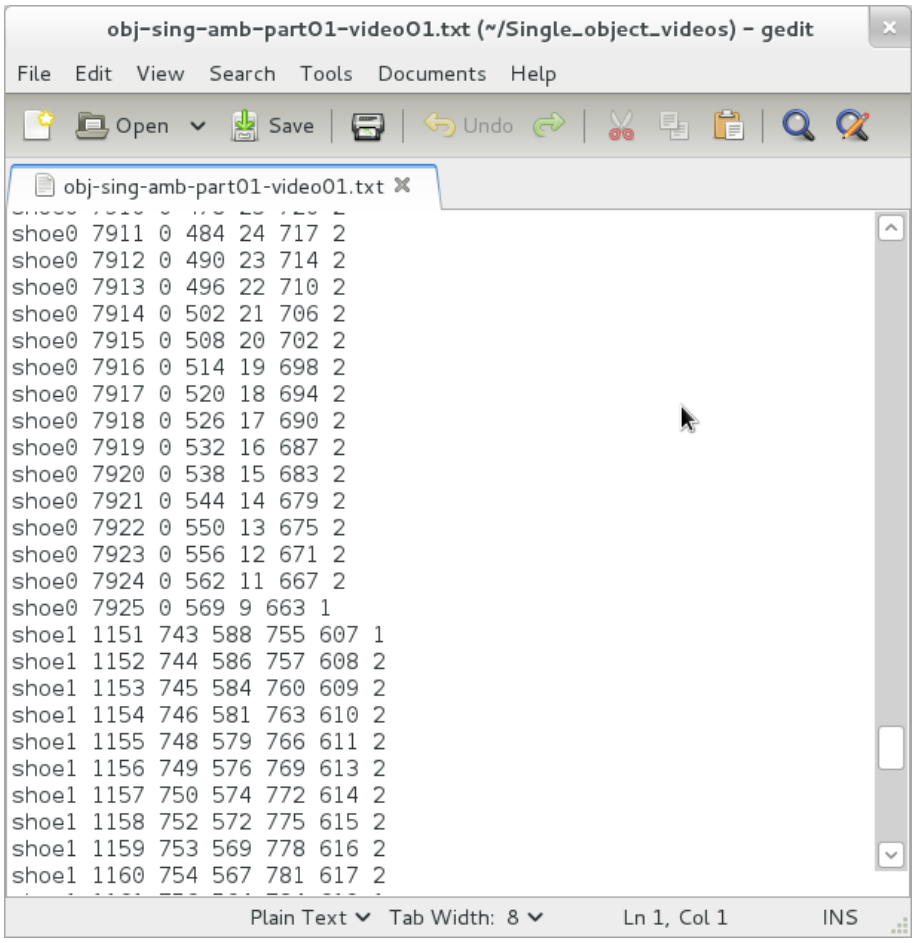
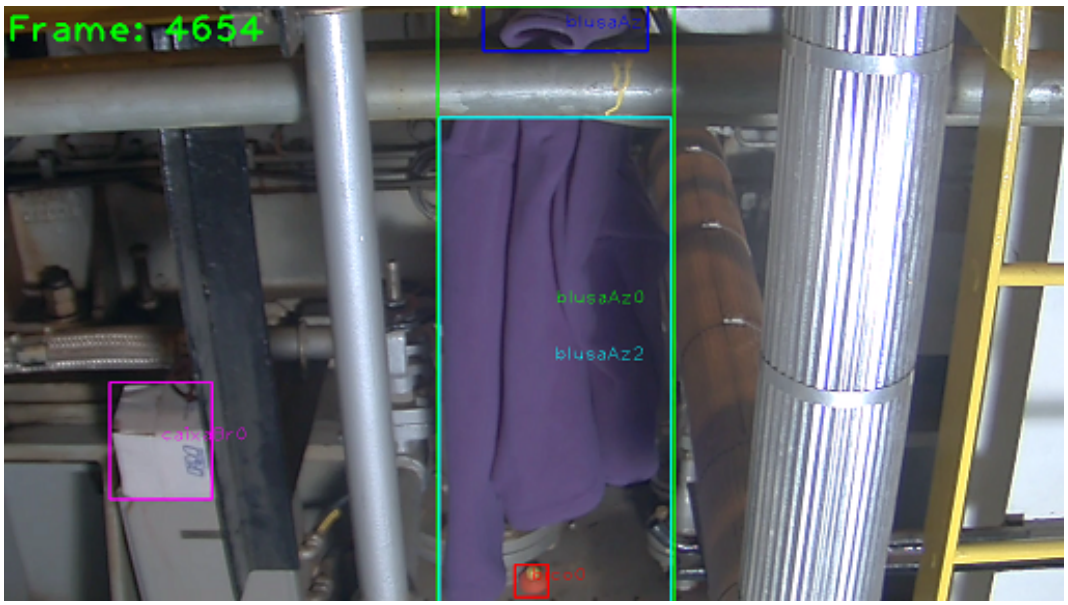


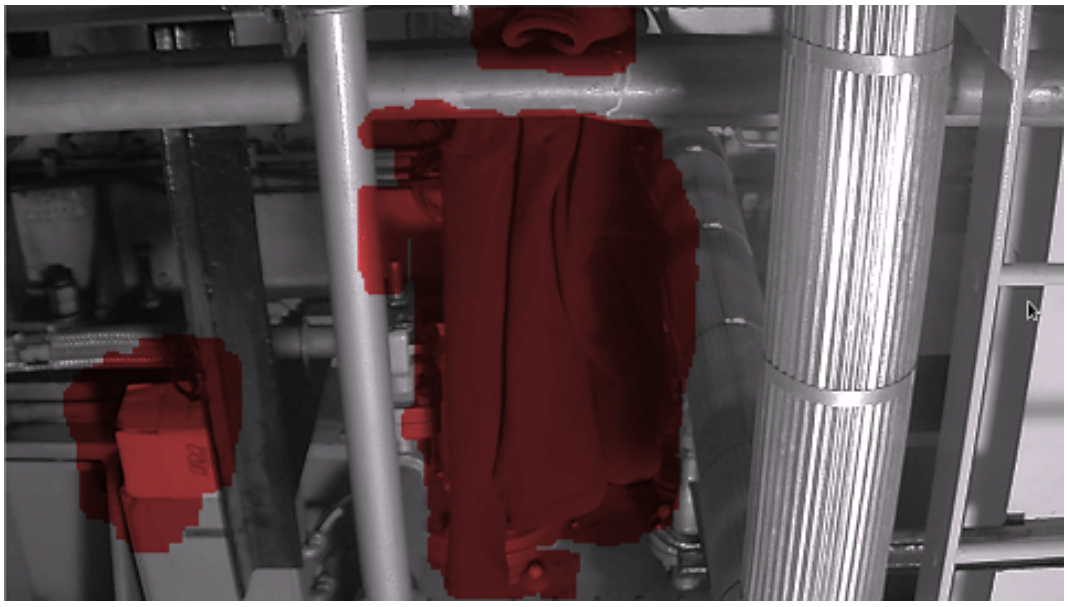
Figure 4.10: Example of a text file excerpt generated by the annotation process.

Figure 4.11: Example of abandoned-object annotation with the help of the developed annotation tool.

A visual example of the results provided by the annotation and detection processes is provided in Fig. 4.12 for a specific multi-object video frame. In this figure, we notice the presence of the 'spotlight box', 'blue coat' and 'bottle cap' objects (shown in Figs. 4.1e, 4.1g and 4.1m, respectively), with the 'blue coat' being partially occluded by a large pipe in the foreground, as properly identified in Figs. 4.12a and 4.12b.



(a)



(b)

Figure 4.12: Example of a multiple-object video frame: (a) annotation result; (b) detection result.

Chapter 5

System Design

This chapter describes in detail the implementation of all the processing steps listed in Chapter 3, including the techniques proposed in this thesis for a reliable, real-time operation of the monitoring system.

5.1 Initial Video Alignment

The initial synchronization between the reference and target videos can be implemented automatically using a maximum-likelihood approach based on the video's motion data. It is in block n.1, shown in Fig. 5.1, that this step is performed.

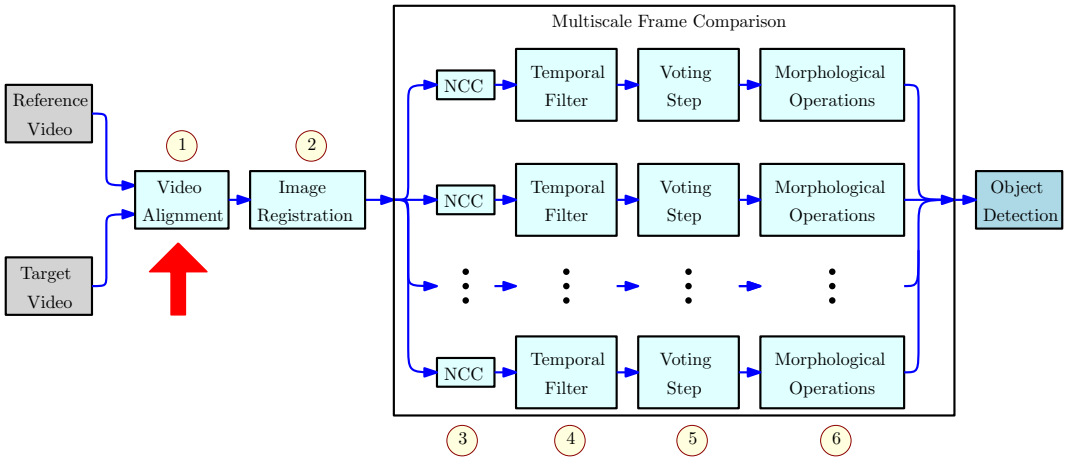


Figure 5.1: The initial video alignment step is performed in block n.1, pointed by the red arrow.

In this scheme, the robot's motion model assumes a constant speed along the straight track, with the direction changing when the robot reaches the track ends. The instantaneous camera speed can be estimated from the homography transformation between consecutive frames, as determined in Section 5.2. By integrating the horizontal component (along the track) of the camera speed, one can obtain the horizontal camera displacement in each frame index n up to a constant δ.

Fig. 5.2 shows camera displacements as a function of the frame index estimated from actual reference (solid curve) and target (dotted curve) video sequences. In this plot, the maxima and minima of each curve can be associated to the two track ends. One should note that different initial positions of the robot give similar curves differing only on their mean values.

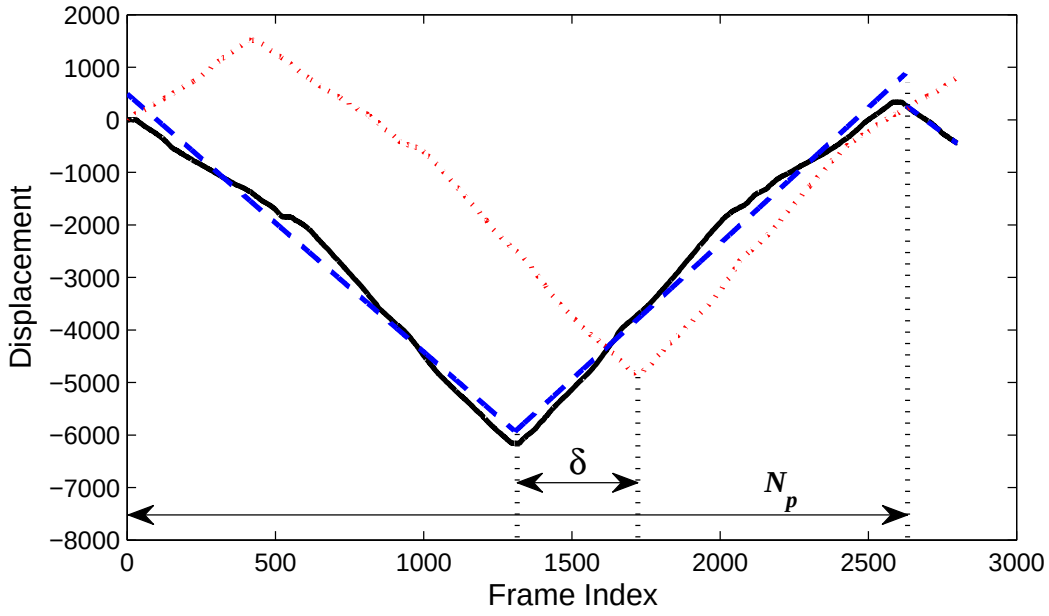


Figure 5.2: Example of camera displacements estimated from reference (solid line) and target (dotted line) videos, where the piecewise-linear dashed line represents the robot's movement with a constant-speed model.

The displacement curves are noisy due to the camera vibration. A noiseless motion model $d_{\mathrm{r}}(n)$, however, can be determined by performing the least-squares fitting of a piecewise-linear model composed of two straight lines of opposite angular coefficients. Such a model for the reference displacement is shown as a dashed line in Fig. 5.2, where, without any loss of generality, the direction change is assumed to be at $n = 0$. A similar motion model $d_{\mathrm{t}}(n)$ can be generated for the target displacement. Once again, since we do not know the initial position of the camera, this function may have an arbitrary average level.

Using the two displacement models, an initial frame alignment between the reference and target videos can be determined as the displacement δ that maximizes the cross-covariance between the $d_{\mathrm{r}}(n)$ and $d_{\mathrm{t}}(n)$, that is

$$\hat{\delta} = \underset{\delta}{\operatorname{argmax}} \left\{ \sum_{n} (d_{\mathrm{r}}(n-\delta) - \mu_{\mathrm{r}})(d_{\mathrm{t}}(n) - \mu_{\mathrm{t}}) \right\}, \tag{5.1}$$

where μ_{r} and μ_{t} are the average values of $d_{\mathrm{r}}(n)$ and $d_{\mathrm{t}}(n)$, respectively.

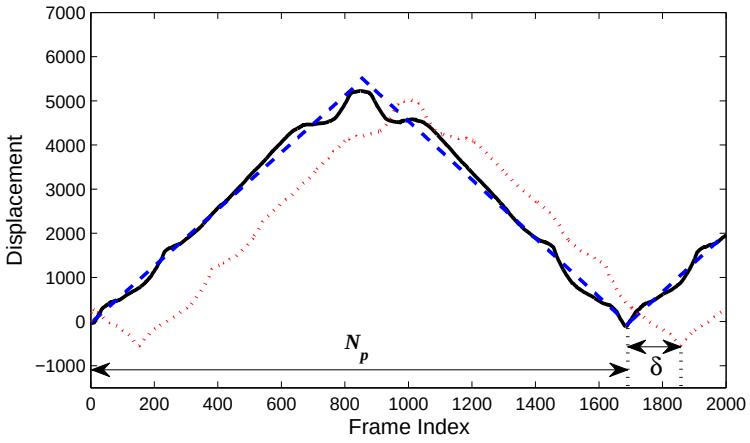
At a first glance, the summation interval in Eq. (5.1) should be approximately

equal to the number of frames N_p of one full back-and-forth movement of the camera. In Fig. 5.2, for instance, $N_p \approx 2600$ frames. However, this would impose severe restrictions for the system's real-time operation, as one would need to record a full back-and-forth cycle before synchronizing the two videos. To mitigate this issue, the summation interval can be restricted to only $\Delta \approx 200$ frames, as long as one guarantees that it contains one change in the camera moving direction to allow a proper pattern matching.

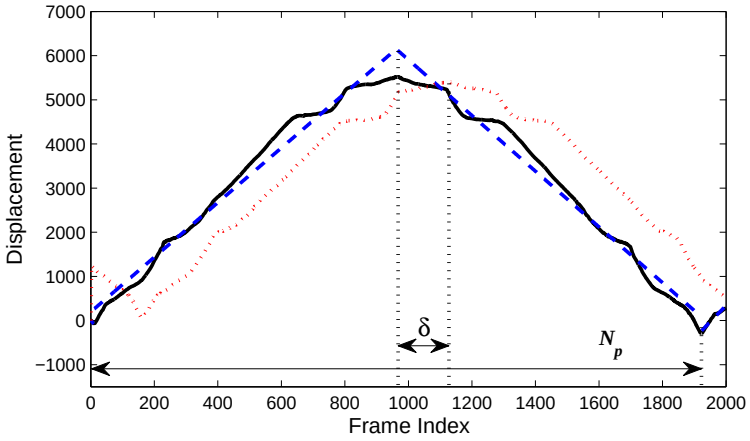
It is important to note that such initial alignment does not need to be extremely precise, since errors of a few frames translate into small displacements that can be compensated for in the image registration stage (see Section 5.2).

This particular video alignment experiment was done with video sequences in which a temporal subsampling of 8 was performed.

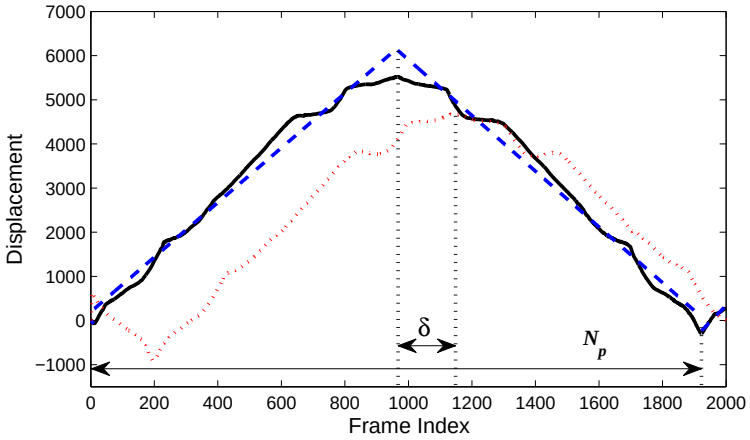
Fig. 5.3 shows more examples of video alignment experiments, using the three videos containing multiple objects.



(a)



(b)



(c)

Figure 5.3: Examples of camera displacements estimated from reference (solid line) and target (dotted line) videos, where the piecewise-linear dashed line represents the robot's movement with a constant-speed model: (a) Example generated with the first video containing multiple abandoned objects and its respective reference video; (b) Example generated with the second video containing multiple abandoned objects and its respective reference video; (c) Example generated with the third video containing multiple abandoned objects and its respective reference video.

5.2 Image Registration

This step of our method is performed on block n.2, shown in Fig. 5.4

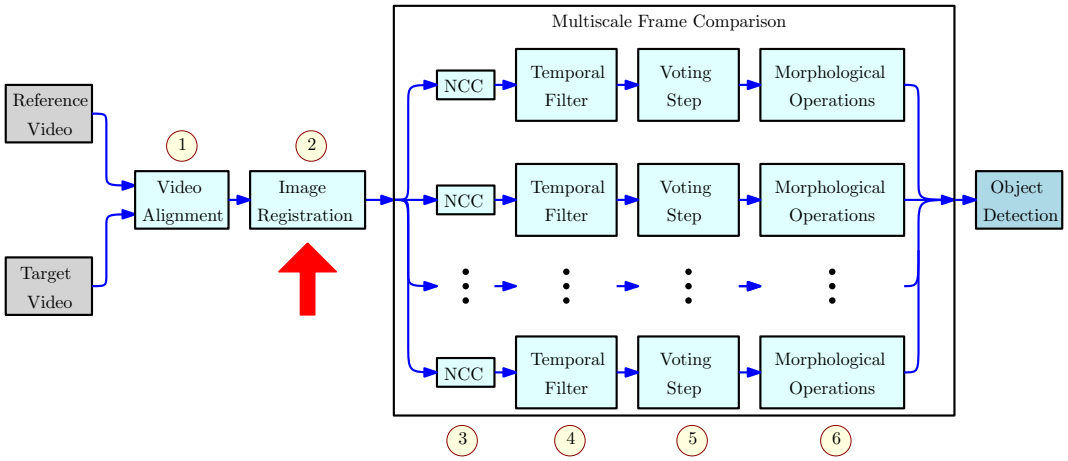


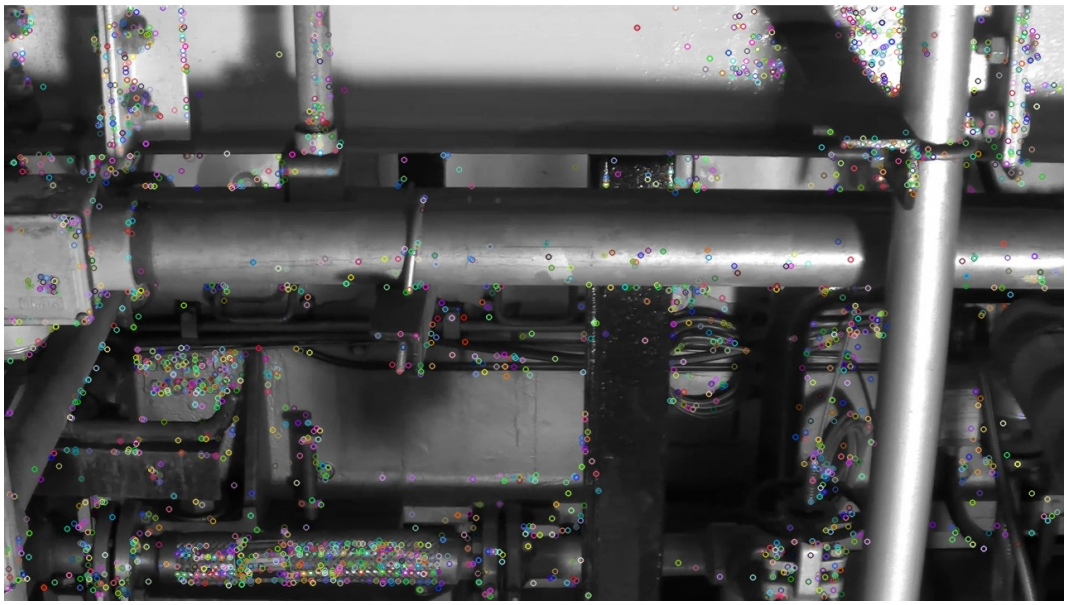
Figure 5.4: The image registration step is performed in block n.2, pointed by the red arrow.

Assuming that the reference and target videos have been initially aligned, as described in Section 5.1, the speeded-up robust feature (SURF) algorithm is used to identify the points of interest (PoI) on two corresponding frames of both video sequences [58]. In the proposed system, the HD video resolution was downsampled by a factor of 4 in each dimension, to reduce the computational complexity, allowing the system to operate in real time. Larger decimation factors would yield smaller images (in terms of pixels) and consequently fewer PoIs. This might both jeopardize the subsequent image-registration stages and erase small objects from the scene.

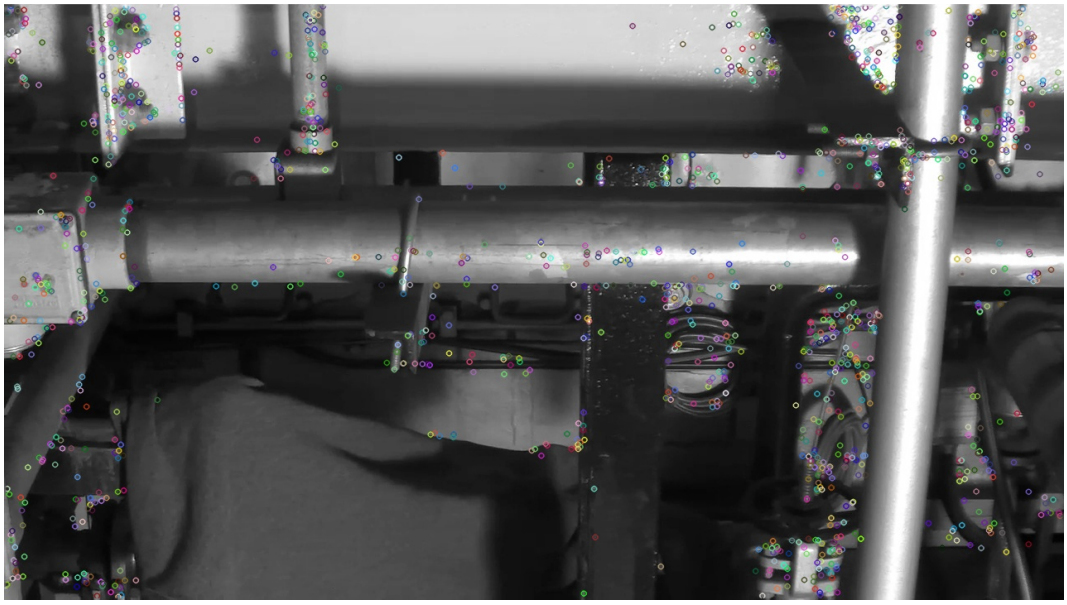
Fig. 5.5 shows an example of PoIs obtained in a reference and a target frames. One might note that flat surfaces generally do not present PoIs.

In [59], using a much simpler version of our abandoned object detection system, presented in [60], the authors assess not only the performance of the SIFT and SURF algorithms, but also of BRISK [61] and FREAK [62], considering their OpenCV 2.4.8 implementations. As the current and the older versions of our system were implemented using OpenCV 2.3.1 (without BRISK and FREAK implementations), the authors in [59] had to make the necessary modifications in order to perform the mentioned study. Based on their conclusions and considerations about these algorithms implementations in the different OpenCV versions used, we decided to employ the SURF method in the current version of our system, as it generates approximately the same amount of PoIs than SIFT, but can be up to 5 times faster (we performed this processing time test with the OpenCV 2.3.1 version).

In a subsequent step, a point-to-point correspondence is determined among the PoI sets from the reference and target frames. The random sample consensus



(a)



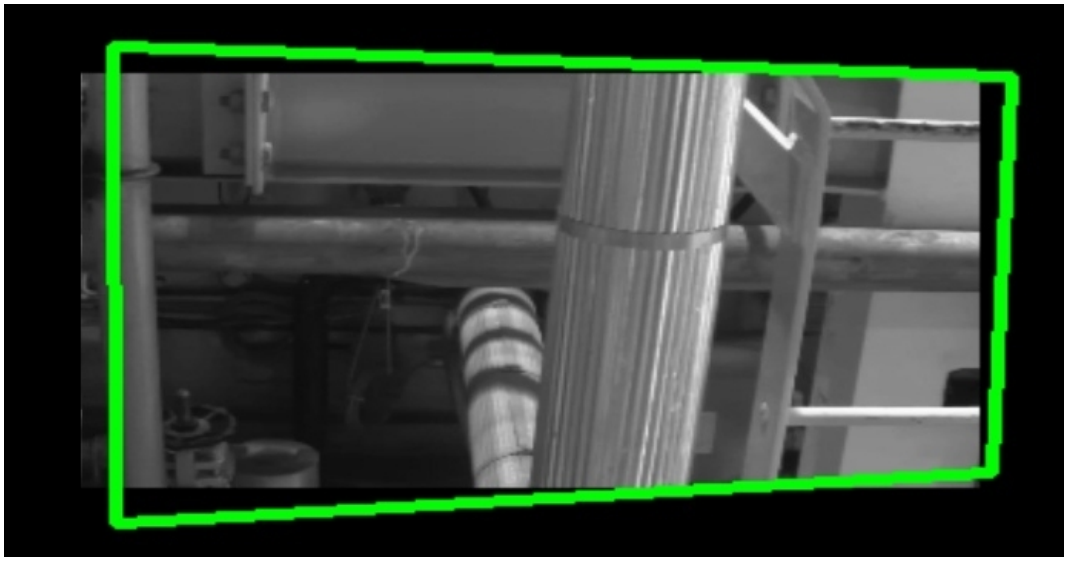
(b)

Figure 5.5: Example of PoIs obtained in reference and target frames: (a) reference frame; (b) target frame.

(RANSAC) algorithm [8, 63], an iterative method used to estimate parameters of a model from a set of observed data containing outliers, is employed to select pairs of corresponding points in the images. Based on these correspondences, the homography transformation that best maps the reference PoI set onto the target PoIs is determined to compensate for any difference in the camera positioning in the temporally aligned reference and target videos [8, 63]. An example of such homography transformation can be seen in Fig. 5.6.

Homographies are invertible transformations of a projective space to itself that

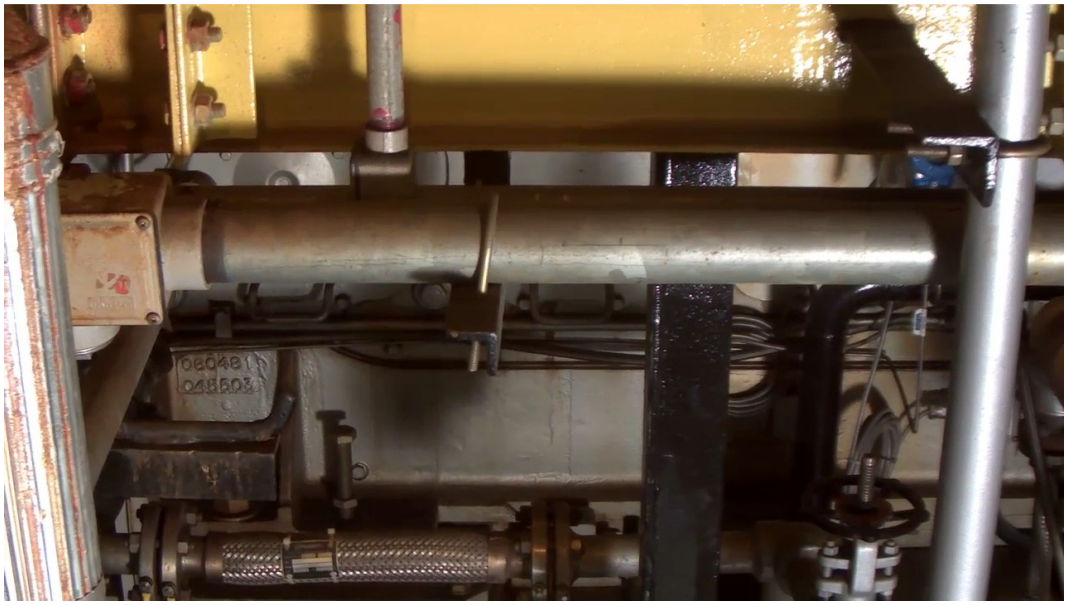
(a)



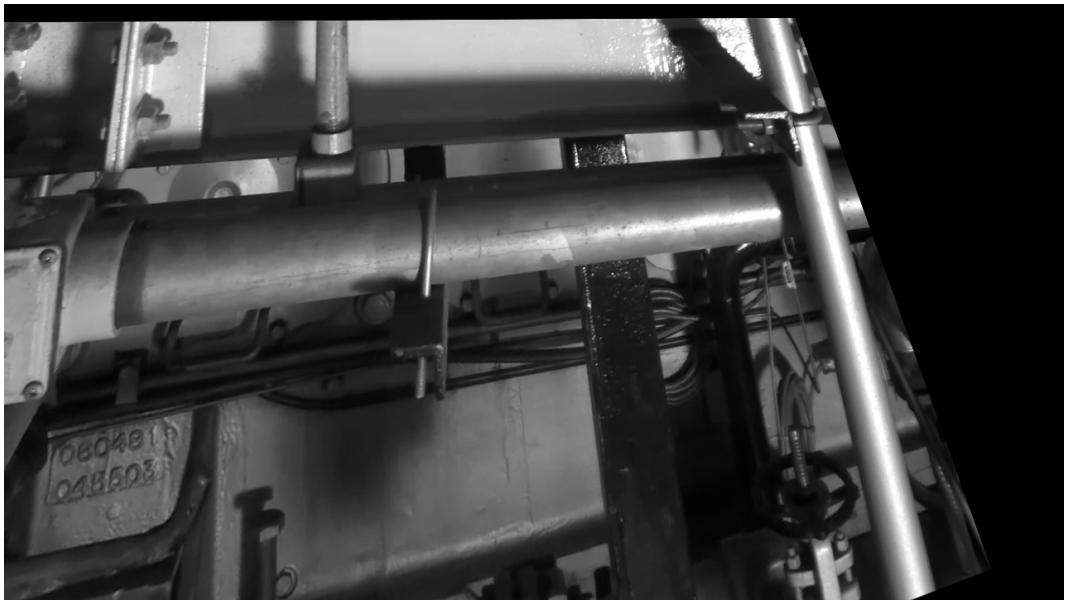
(b)

Figure 5.6: Example of homography transformation. The green quadrilateral shows the image-plane mapping from the first image into the second one: (a) reference video frame; (b) target video frame.

maps straight lines onto straight lines. In our case, the image obtained from the reference video is mapped to the image from the target video. To calculate a homography, it is necessary to have at least 4 correspondence points, with no 3 of them being collinear. However, the more points are used, the more robust is the obtained result, as more points can help to better cope with measurement errors, for example. To further illustrate the concept of homography transformation, an example of image warped after a homography was apllied to it can be seen in Fig 5.7.



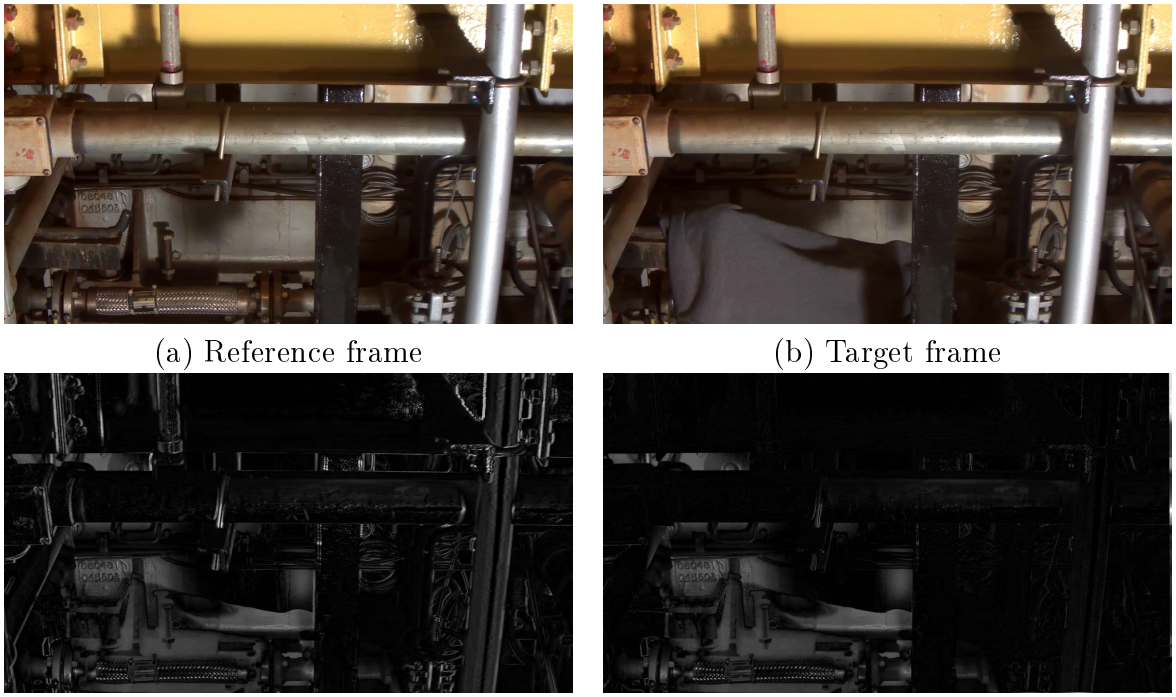
(a)



(b)

Figure 5.7: Example of image warped after a homography was apllied to it: (a) image before homography was applied to it; (b) warped image, after homography was applied to it.

Fig. 5.8 illustrates the effect of the geometric registration process using the absolute difference between two frames. One may note that, in the image generated by the absolute subtraction between the frames in which the geometric registration procedure was performed, there are more dark regions, that is, there are less differences between the registered frames.



(a) Reference frame

(b) Target frame

(c) Absolute difference between frames without geometric registration

(d) Absolute difference between frames with geometric registration

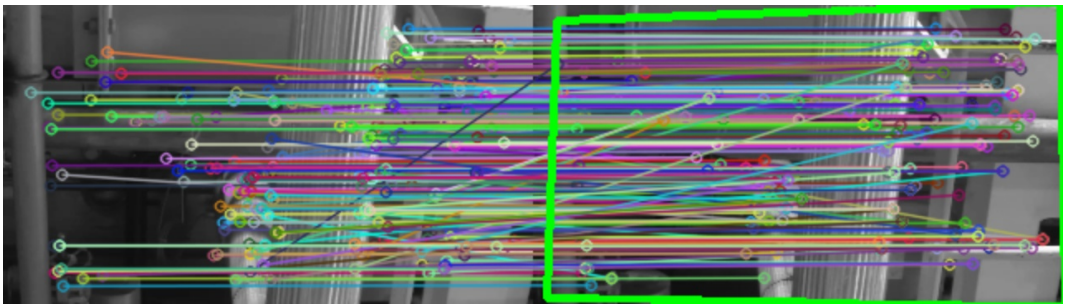
Figure 5.8: Example of the effect of the geometric registration process. In this image, one can also notice the considerable difference in the images promoted by the shadow cast by the coat.

There is a depth dimension in the video sequences, as they show a surveillance scenario. As abandoned objects can be at different depths, as they map straight lines to straight lines, homographies may cause some parallax issues. However, as we have a real-time constraint, and as homography calculations are not too computationally costly, we decided to keep using them in our image registration step and cope with the small parallax effects that may happen.

In this work, considering the camera's horizontal movement, all reference-target PoI correspondences yielded by the SURF algorithm with an angular displacement with absolute value larger than 1° can be immediately discarded. This strategy not only reduces the computational complexity associated to the RANSAC outlier removal procedure, but also improves the consistency of the resulting homography transformation.

An example of the improved consistency mentioned above can be seen in Fig. 5.9, where one can notice that, in the example without our angular restriction, the

resulting homography does not correspond to the horizontal movement performed by the moving platform.



(a)

(b)

Figure 5.9: Example of homographies generated with and without our angular restriction: (a) without angular restriction; (b) with angular restriction.

5.3 Multiscale Frame Comparison

Once the two videos are registered, one would be tempted to employ plain frame subtraction to detect any objects present in only one of the video sequences. However, this scheme tends to fail in cluttered environments, as seen in Fig. 5.10. The reason for this is twofold. First, there are usually objects at many different depths, and a homography can only perform an accurate registration if the objects are all in the same plane. Also, the excessive amount of image details involved would require very precise registration. For that matter, the normalized cross-correlation (NCC) [8] between the two images is often employed, followed by a simple threshold detection, which yields a binary image indicating areas of the target frame that are candidates to contain abandoned objects. In addition to that, in this study we propose to apply some spatio-temporal post-processing on the binary masks generated.

The NCC $k(m,n)$ between the images $r(m,n)$ and $t(m,n)$ over a window $\mathcal{W}(m,n)$ centered in the pixel position (m,n) can be defined as

$$k(m,n) = \frac{1}{N_w} \sum_{(m',n') \in \mathcal{W}(m,n)} \frac{[r(m',n') - \bar{r}][t(m',n') - \bar{t}]}{\sigma_r \sigma_t}, \tag{5.2}$$

(a)

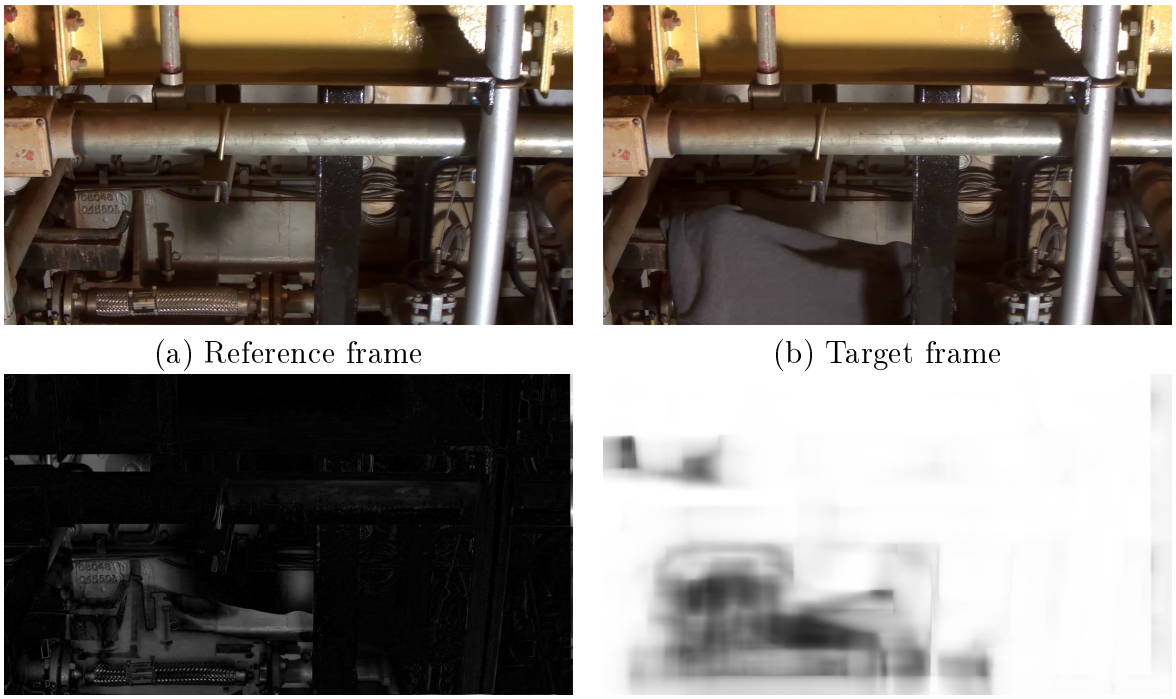
(b)

(c)

Figure 5.10: Example of plain frame subtraction: (a) Reference frame without abandoned object in it; (b) Target frame with abandoned object in it (backpack); (c) Frame generated by performing the absolute subtraction between reference and target frames.

where N_w is the total number of pixels in the window $\mathcal{W}$, $\bar{r}$ and $\bar{t}$ are the average values of r and t inside the window $\mathcal{W}(m,n)$, respectively, and σ_{r} and σ_{t} are their respective standard deviations.

Fig. 5.11 illustrates the difference in performance between the absolute subtraction between registered frames and the production of a NCC image using such frames. It is clear that it is in the NCC image that the region where the abandoned object is stands out the most.



(a) Reference frame

(b) Target frame

(c) Absolute difference between frames with geometric registration

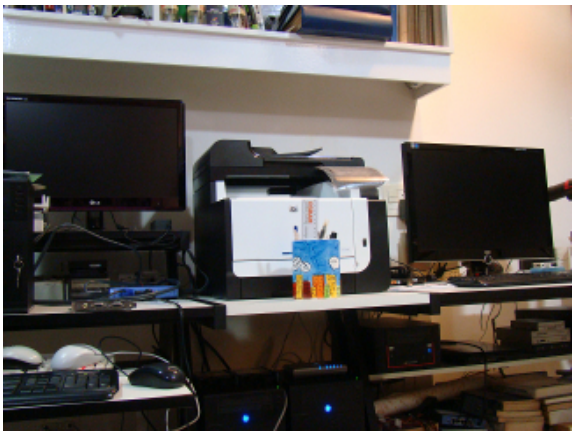
(d) NCC image without threshold detection

Figure 5.11: Example showing the differences between plain subtraction between registered frames and the production of a NCC image. In the NCC image, one can once again notice the difference in the frames promoted by the shadow cast by the coat.

The NCC window size should be in the same order of the apparent size of the abandoned object to be detected (as illustrated in Fig. 5.12, obtained with a static camera), which is considered unknown or may even vary if more than one object appear on the same frame, as illustrated in Fig. 5.13. In fact, large windows tend to overlook small objects, whereas small NCC windows may identify a single large object as several small ones. Therefore, for a robust detection, one must compute the NCC function between two frames with different window sizes, what may greatly increase the computational complexity of the resulting algorithm.

A proposed solution to this problem is to perform a multiscale NCC computation, which employs a fixed window ($K \times K$ pixels) on several downsampled versions of the reference and target videos. The whole multiscale procedure starts with a frame downsampling factor of 64, which greatly simplifies the NCC computation and makes the fixed window suitable to detect larger objects. Progressively smaller objects are then searched for by using the same $K \times K$ window size with increasing resolution

(a) Reference frame



(b) Target frame

(c) NCC image generated with small (3×3) window

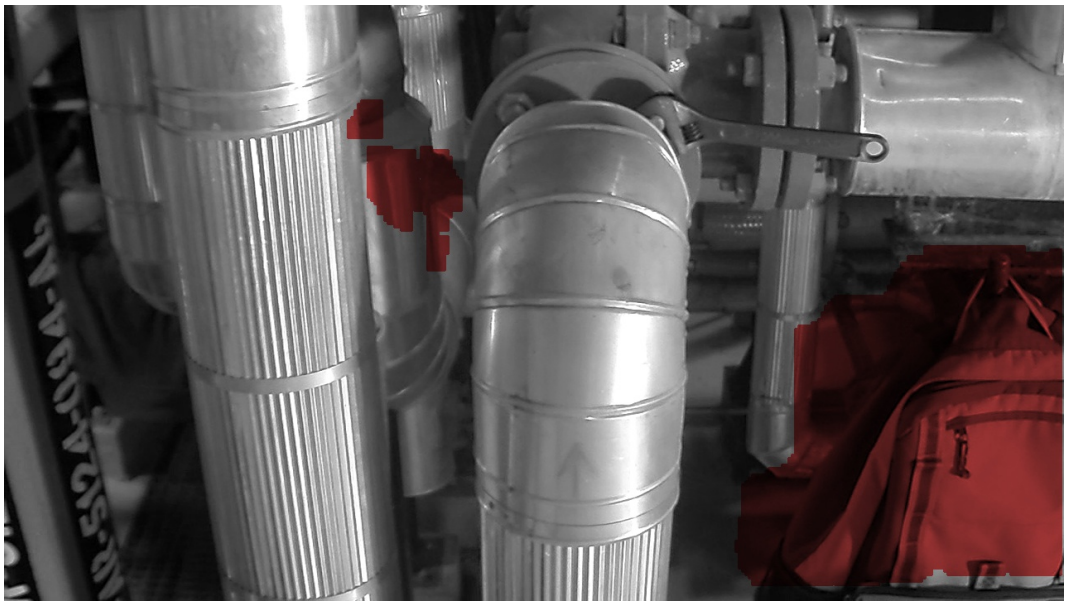
(d) NCC image generated with window compatible with abandoned object's dimensions

Figure 5.12: Example illustrating the effect of different NCC window dimensions. In image (d), one can also notice that, besides the abandoned object, reflections in the monitor screens and in the upper right corner of the wall were detected.

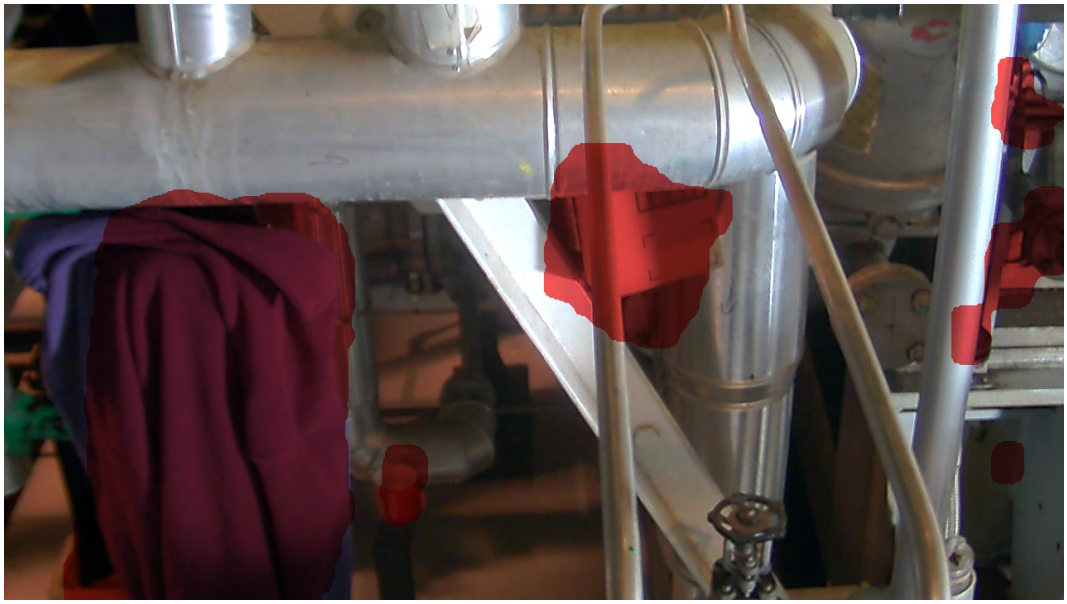
images. If a larger object is detected, the NCC computation in the corresponding region can be skipped in higher image resolutions, thus reducing the computational complexity even further.

Figure 5.13: Example of frame in which objects of different sizes appear. Partial occlusion can make a larger object (green box and umbrella in this example) look like it is in fact one or more smaller objects.

Due to the real-time constraint, one has to restrict the allowed values of K, image resolutions, and downsampling factors. The value of K is set based on the size of the larger object to be detected in the smallest resolution to be used. For the employed database, the NCC window size was set to $K = 5$. A large K leads to missed detections due to low NCC values, as the missed wrench in Fig. 5.14a. On the other hand, a small K highly increases the NCC sensitivity, yielding false positive detections, as exemplified in Fig. 5.14b.



(a)



(b)

Figure 5.14: Adjustment of NCC window size K (red stains indicate abandoned-object detection): (a) Excessively large values of K tend to oversee smaller objects (false negatives) such as the wrench at the top right; (b) Excessively small values of K increase the sensitivity of the NCC measure, leading to false positives.

The number of scales to be employed is defined by the operator, considering the variation in the apparent sizes of the abandoned objects that one wants to detect. For the VDAO database, four different image resolutions were employed, corresponding to image downsampling factors, in each direction, of 64 (suitable for the detection of larger objects), 32, 16, and 8 (smaller objects), leading to a satisfactory overall system performance, as illustrated in Fig. 5.15. Additional image resolutions could fit even larger or smaller objects at an additional cost on the resulting computational burden. With the NCC window size K fixed, this increase in computational complexity is linear with the number of scales.

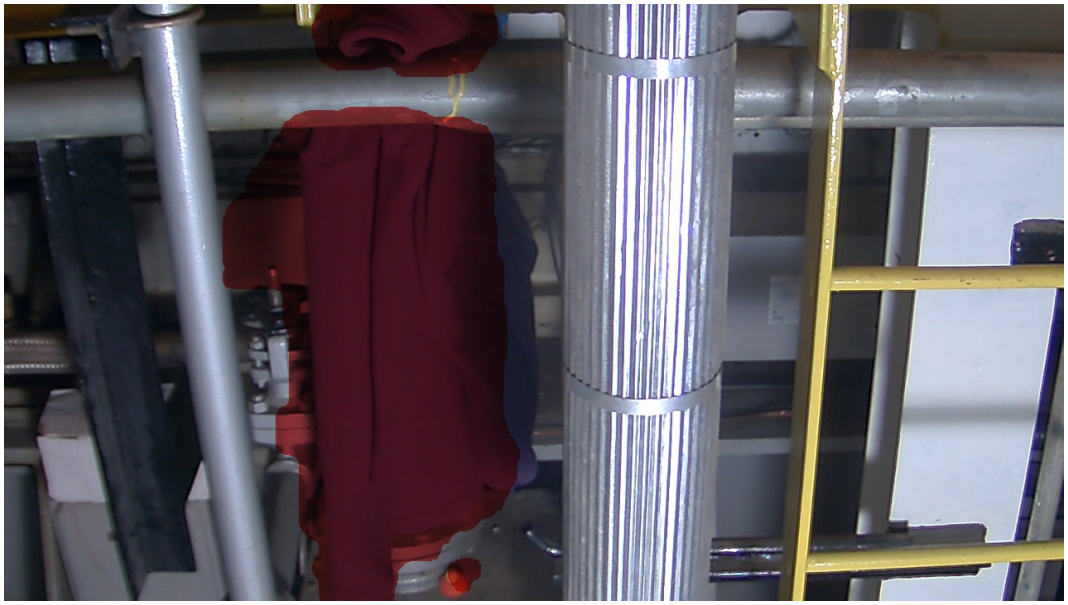


Figure 5.15: Example of VDAO-database objects of different sizes being detected with the multiscale approach: The large coat is detected with the video decimation factor of 64, whereas the small bottle cap is detected with the decimation factor of 8.

The binarized NCC image generation step, in our multiscale context, is generated in block n.3, as shown in Fig. 5.16.

Figs. 5.17 and 5.18 exemplify our multiscale approach.

Fig. 5.17a shows the target image with the pink backpack, umbrella, blue box and bottle as abandoned objects. In Fig. 5.17b, the lowest resolution, only the largest object (pink backpack) is detected. The higher the resolution, the more objects start to be detected. In Fig. 5.17e, the highest resolution in this example, even the shadow cast by the bottle is detected. One can also notice in this example that the umbrella and the blue box are detected as a single object. This effect is the result of the square dimensions of our NCC window together with the memory effect of our voting step and the morphological closing operation. Both the voting step and the morphological operations will be explained in Section 5.4.

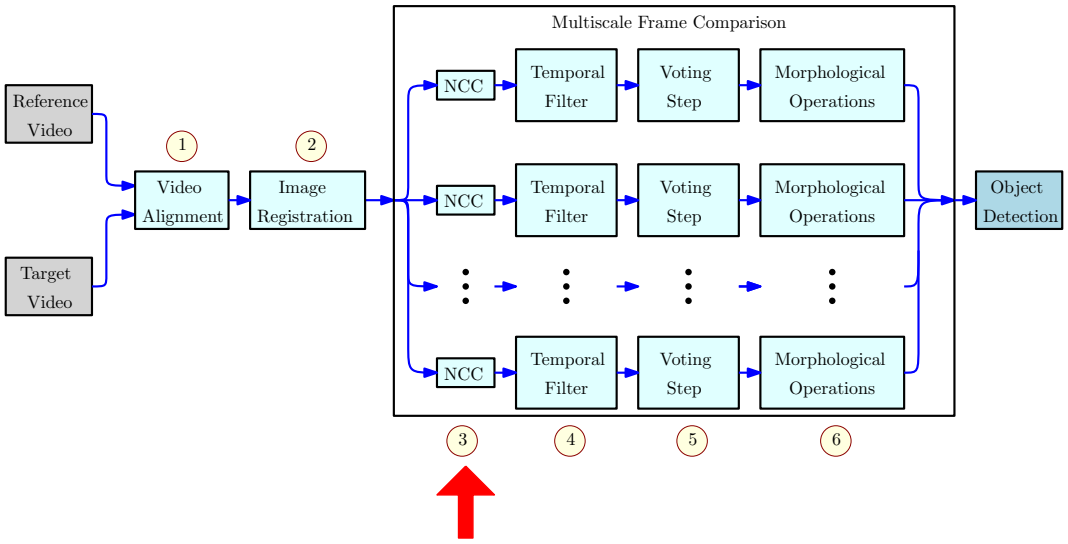
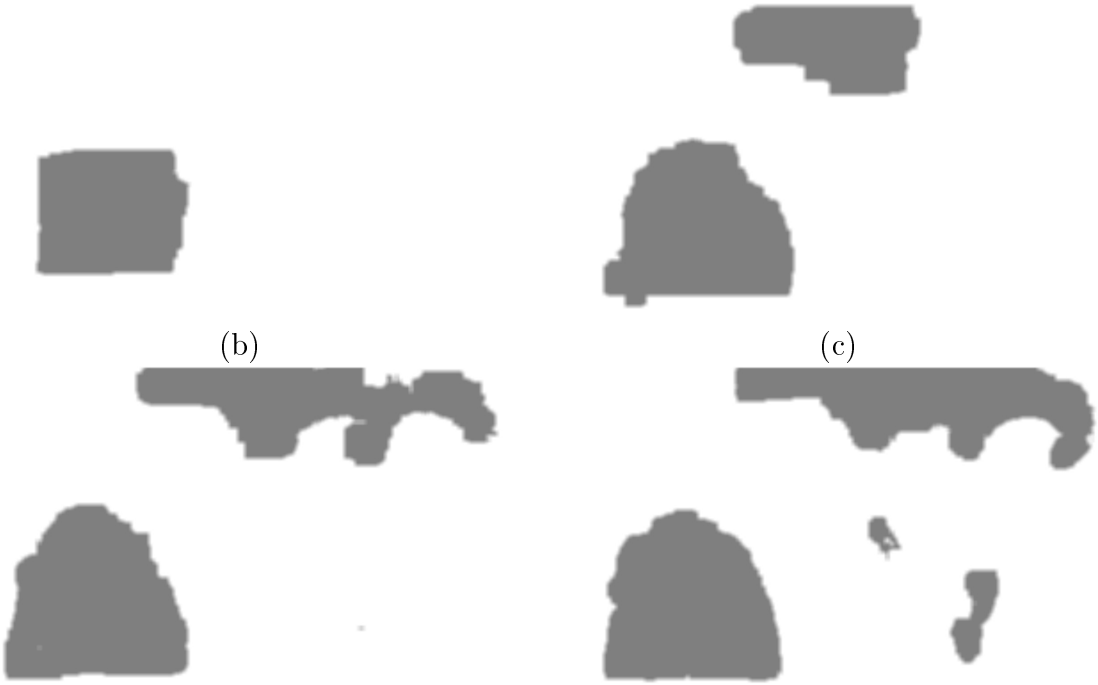


Figure 5.16: The binarized NCC image generation step is performed in block n.3, pointed by the red arrow.

Fig. 5.18a shows the target image with a backpack (large object), a string roll (medium object) and a mug (small object). In Fig. 5.18b an image that is subsampled by 64 is used, and only the largest object, the backpack, is detected. In Fig. 5.18c an image subsampled by 32 is used, and the string roll is also detected. In Fig. 5.18d an image subsampled by 16 is used and the smallest object, the mug, is also detected. Note that, at this resolution, a backpack strap is also detected.

(a)



(b) (c)

(d) (e)

Figure 5.17: Example of our the multiscale approach: (a) Target frame; (b) Largest object (pink backpack) detected in the lowest frame resolution; (c) In a higher frame resolution, the blue box and the umbrella start to be detected; (d) In an even higher frame resolution, one can see the umbrella being properly detected; (e) In the highest frame resolution, the bottle (and its shadow) start to be detected.

(a) Target image

(b) Image subsampled by 64

(c) Image subsampled by 32

(d) Image subsampled by 16

Figure 5.18: Example of detection masks using the multiscale approach: in image (b), only the largets object (backpack) is detected; in (c), the string roll is also detected; finally, in (d), the mug algo starts to be detected. The green box is not detected in this example because it dit not appear a sufficient number of times for it to be detected yet.

5.4 Detection Mask Post-Processing

In order to reduce both false positive (spurious) and false negative (missed) detections, three additional processes are sequentially performed onto the resulting pixels of the NCC multiscale masks: a temporal filtering procedure, a voting strategy and opening-closing morphological operations.

First, temporal filtering is applied to the NCC mask frame sequence [8] such that

$$M(m,n,p) = \prod_{i=0}^{L_{\text{tf}}-1} \hat{k}(m,n,p-i), \tag{5.3}$$

where p is the frame index, L_{tf} is the temporal-filter length and $\hat{k}(m,n,p)$ is a binary version of the NCC output $k(m,n,p)$. This binary mask is defined by a threshold b_t (see Section 6.1). This filtering operation requires a registration procedure among nearby frames. This can be done using a homography such as the one described in Section 5.2. This filtering stage is ideal to remove most of the false positive occurrences from the detection mask, as it requires the intersection of L_{tf} consecutive

masks to activate a given pixel, as seen in Fig. 5.19. An excessively large value of L_{tf}, however, tends to produce false negatives in our detection process. This is particularly relevant in our case since the camera is moving, and large displacements among frames that are too distant in time tend not to be dealt with properly with a homography. In our setup, the temporal filter length was set to $L_{\text{tf}} = 5$. This choice is justified in Section 6.2.

(a)

(b)

Figure 5.19: Example of false positive elimination by the temporal filtering stage: (a) Detection mask produced without temporal filtering; (b) Detection mask produced with temporal filtering.

In our multiscale context, the temporal filtering step is performed in block n.4,

as shown in Fig. 5.20.

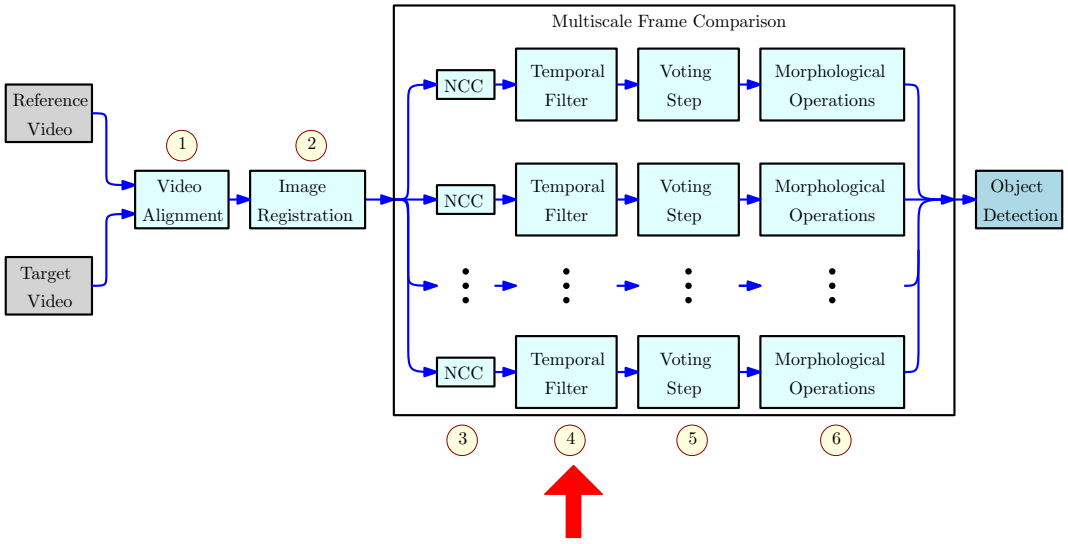


Figure 5.20: The temporal filtering step is performed in block n.4, pointed by the red arrow.

Following the temporal filtering, a voting procedure is employed to increase the detection robustness. The rationale for the voting procedure is that it is unlikely that an object will disappear from the video for just a few frames to reappear again later. Likewise, it is unlikely that an object will appear in the video for just a few frames. In the voting step a new mask $M_v(m,n,p)$ is generated as follows:

$$M_v(m,n,p) = \begin{cases} 1, & \text{if } \sum_{i=0}^{V-1} M(m,n,p-i) \geq v_t \\ 0, & \text{if } \sum_{i=0}^{V-1} M(m,n,p-i) < v_t \end{cases}, \tag{5.4}$$

where $M(m,n,p)$ is defined in Eq. (5.3), V is the voting interval length and v_t is the voting threshold. Once again, this procedure assumes a proper registration of the consecutive masks using a homography transformation as given in Section 5.2. The values of V and v_t determine the minimum amount of times a masking pixel must be activated to be recognized as part of an abandoned object. In practice, these parameters depend on the number of frames a given abandoned object appears in the target video, that is related to the camera speed. The choice of these parameters will be investigated in Section 6.3. Fig. 5.21 shows that the voting step is performed in block n.5.

One could argue that the temporal filter (Eq. (5.3)) and the voting procedure (Eq. (5.4)) are somewhat redundant. However, there are several cases when both are necessary. Generally speaking, the temporal filter and the voting stage work in tandem to eliminate most of false positives from the detection scheme. One

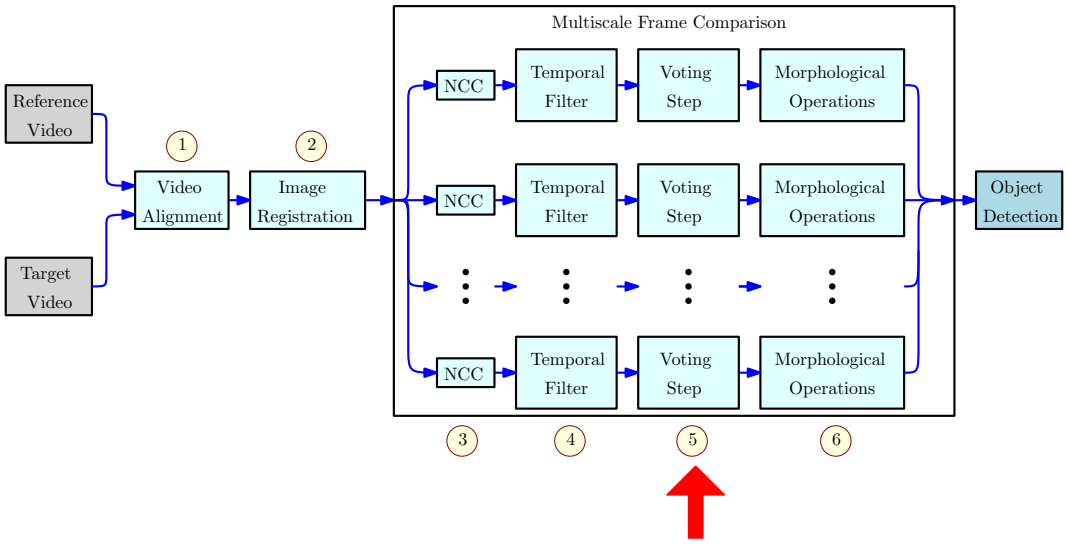
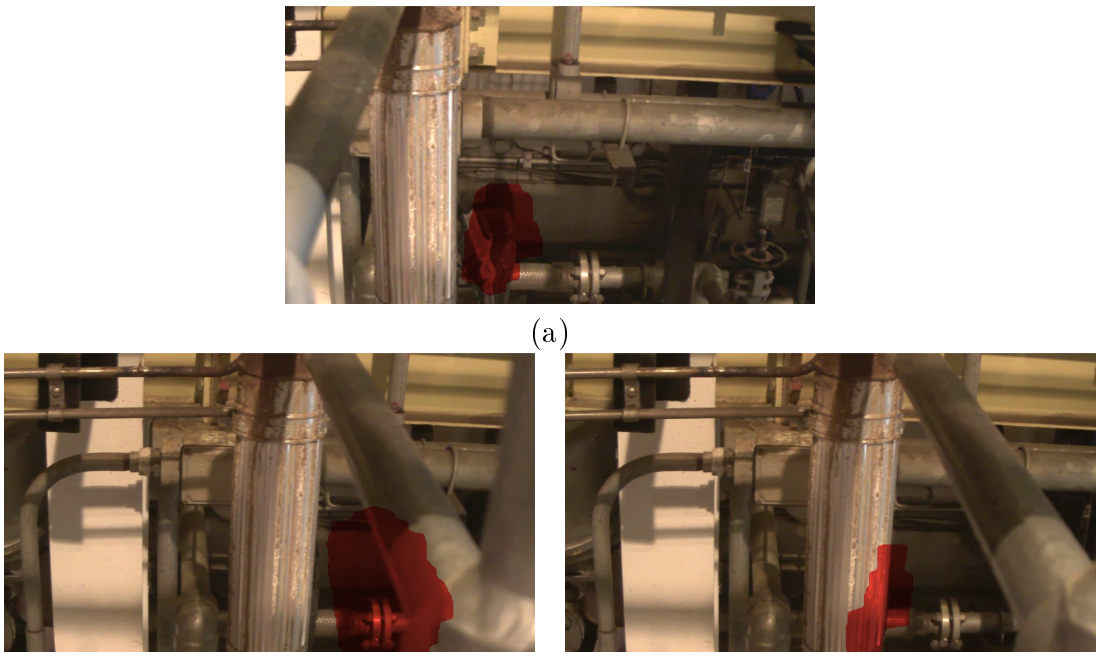


Figure 5.21: The voting step is performed in block n.5, pointed by the red arrow.

illustrative example, depicted in Fig. 5.22, is when partial occlusions generated by foreground obstacles cause an abandoned object to appear in only a limited number of frames. These situations enforce upper limit values for the voting interval length V and its corresponding threshold value v_t. Therefore, they require the temporal filter to remove a priori most of the isolated spurious detections. Figure 5.23 shows another example of a false positive that could have been eliminated if the temporal filtering step were performed.



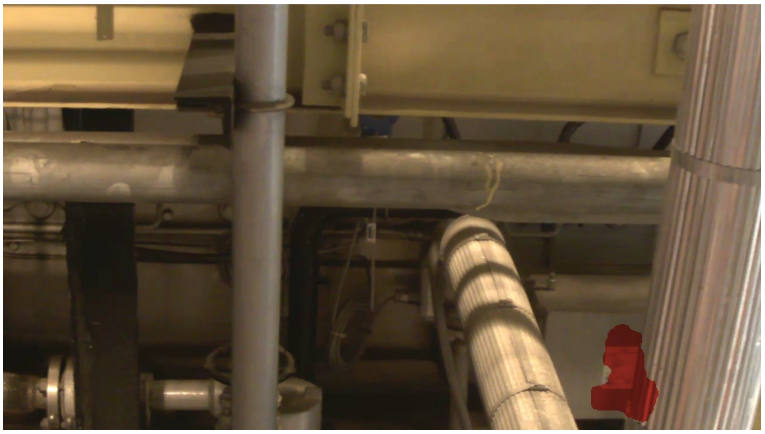
(a)

(b) (c)

Figure 5.22: Example of false positive promoted by occlusion without the temporal filtering step: (a) Initial detection of abandoned object without the temporal filtering step; (b) Displacement of detection mask caused by the foreground pipe in the absence of the temporal filtering (abandoned object is hidden behind bright pipe); (c) Detection mask properly placed by the temporal filtering, which removes the effect from the foreground pipe.

(a)

(b)



(c)

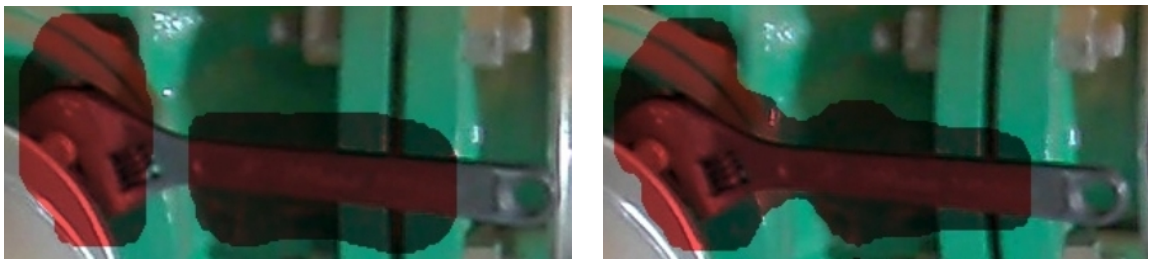
Figure 5.23: Example of false positive promoted by beveled pipe in experiment without the temporal filtering step: (a) Example of target image; (b) Preliminary detection mask generated after the NCC image binarization process relative to target image above, employed directly in the voting step; (c) False positive not eliminated by the less restrictive voting step. Without the temporal filtering procedure, the (re)inserting detection pixels feature of the voting step applied in the many inconstant preliminary detection masks produced by the beveled pipe generates a false positive.

In some cases, after the temporal filtering and voting steps there remain isolated small regions in the detection masks. For removing such pixels, one can perform a morphological binary opening operation. Also, there are cases when the same object is detected by more than one separate mask. In this case, the masks can be connected by a morphological binary closing operation. The binary opening and closing operations can be respectively defined as [64]:

$$A \circ B = (A \ominus B) \oplus B, \tag{5.5}$$

$$A \bullet B = (A \oplus B) \ominus B, \tag{5.6}$$

where A is a binary image, B is the structuring element, $\oplus$ denotes dilation (expansion of the input image by the structuring element B), and $\ominus$ denotes erosion (contraction of the input image by B). The effect of the closing operation is illustrated in Fig. 5.24. In the proposed system, the closing operation is applied to the output of the opening operation.



(a) (b)

Figure 5.24: Example of morphological closing (detail): (a) Before; (b) After. It joins two separate masks that refer to the same object.

The size of the structuring element of the opening operator should be slightly larger than the expected size of the isolated regions that may be present on the detection mask but cannot be larger than the smallest apparent size of the object intended to be detected. In contrast, for the closing operation, the structuring element size should be slightly larger than the expected size of the gaps that separate the disconnected detection masks that may be associated to the same object. In our system, we employed circular-shaped structuring elements with the same radius for the two operations. In full resolution, this radius should be approximately twice as large as the NCC window, and should keep its proportion to the resolution as it decreases. Fig. 5.25 shows that the opening and closing morphological operations are performed in the last block before the final object detection, block n.6.

In Fig. 5.26, one can notice the effect of both morphological operations. These effects are clearer in Figs. 5.26d and 5.26e, as one can notice a small white area inside the largest black spot being covered (effect caused by the closing operation), the largest and the medium black spots being united in one (effect also caused by

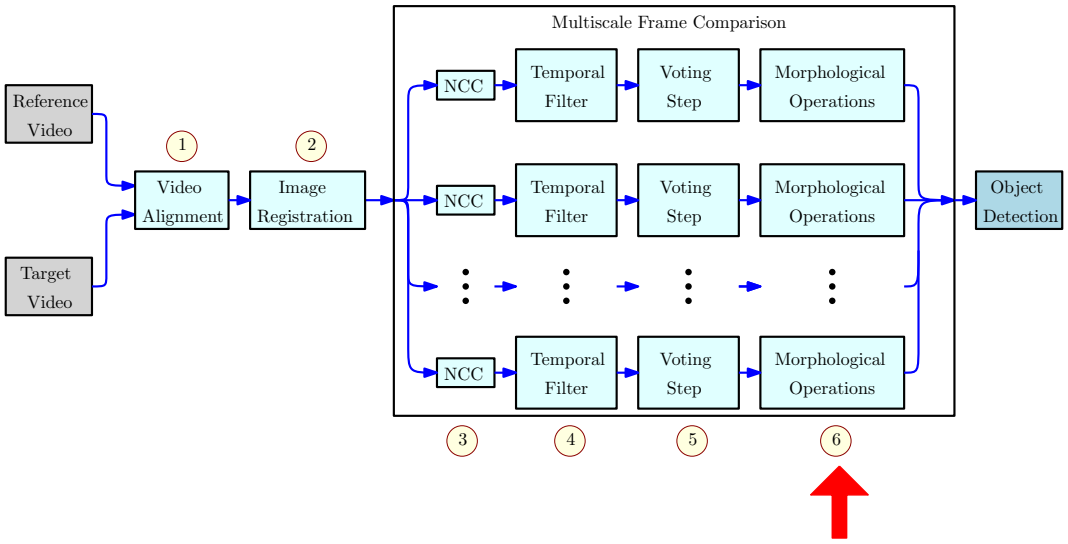
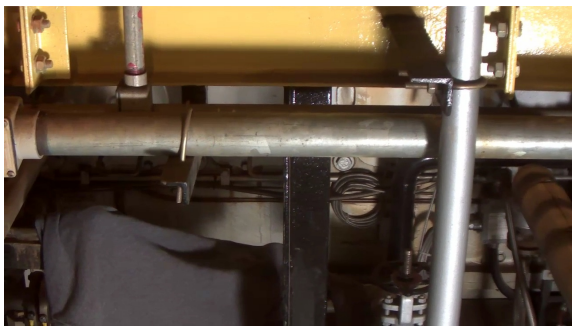


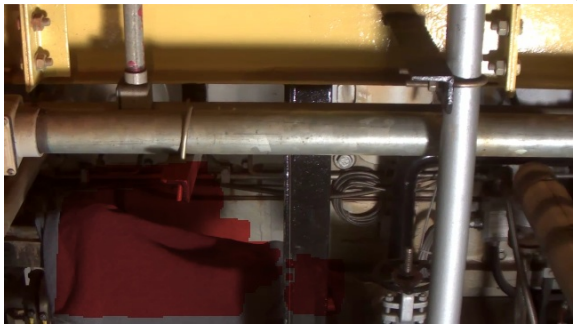
Figure 5.25: The morphological operations are performed in block n.6, pointed by the red arrow.

the closing operation), and the smallest black spot disappearing (effect promoted by the opening operation).

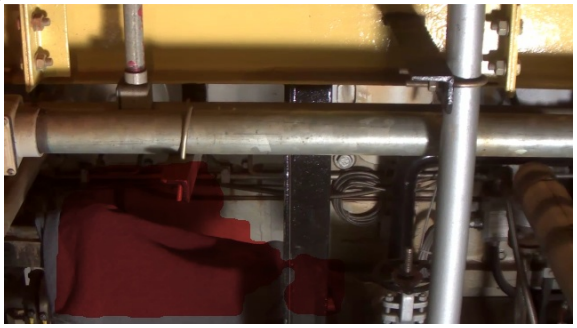
Finally, we perform the union of the detection masks obtained in all resolutions to generate a single, final detection mask.



(a)



(b)



(c)

(d)

(e)

Figure 5.26: Example of morphological operations: (a) Target frame; (b) Abandoned object detection without morphological operations applied to the detection masks; (c) Abandoned object detection with morphological operations applied to the detection masks; (d) Final detection mask generated without morphological operations; (e) Final detection mask generated with morphological operations.

Chapter 6

Tuning of the System Parameters

This chapter outlines the design of all stages described in Chapter 5 aiming at an overall robust performance for the proposed system. As described in Chapter 4, we use the VDAO database [54] for this purpose. In order to do so, we divide the target VDAO sequences in three sets: the training set comprises 16 single-object videos (half with and half without the use of the spotlight), the validation set 38 single-object videos and the testing set with 3 videos with multiple abandoned objects.

The 16 sequences on the training set are used in this chapter to develop routines for the adjustment of each parameter, and determine their values for the VDAO database. The effectiveness of these routines will be assessed in Chapter 7 using the validation set, and the overall algorithm performance will be assessed using the testing set.

To evaluate the effects of a given parameter in our method, several performance analysis curves are derived. In these curves, the true positive measure is determined as the percentage of the bounding box area of the abandoned objects which is covered by the detection mask, and the false positive measure is given by the percentage of remaining frame area covered by detection masks.

To generate each point on the curve, we first calculate, for a given video, the average of the values obtained with each frame where the object and/or a detection spot appears. Then, we calculate the average and the standard deviation of these averages, considering all the objects in the training set. Some aspects, however, affect the performance of the proposed system, and should be taken into account by the reader when considering these measures:

- The voting operation requires a minimum number of frames to detect an abandoned object. This is usually not a problem, since the camera speed is such that there is a large number of frames between the entry of the object in the camera's field of view and its departure. However, when an object is occluded between entering and leaving the field of view, there may not be enough

frames to ensure its detection, which may cause false negative (missed) detections. Fig. 6.1 shows an example of object detection only after it has appeared in a sufficient number of frames in the video sequence.

- A dual problem occurs when the voting strategy keeps the detection mask active even after the object disappears from the scene, artificially increasing the number of false positive detections.

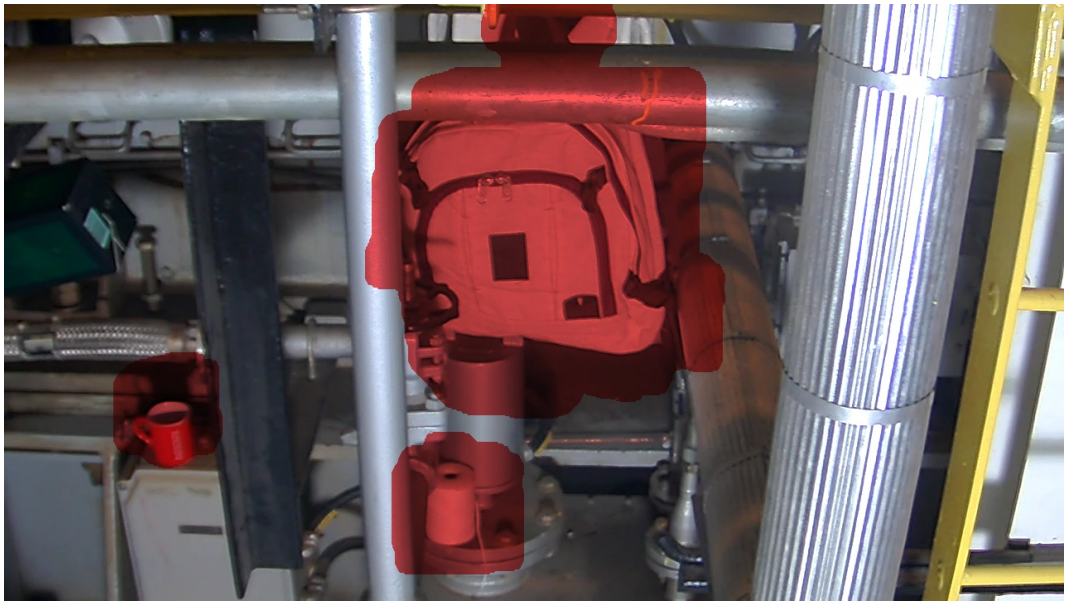
- As the abandoned object does not occupy its entire bounding box, there may be a significant increase on the pixel-level false negative measure.

- Abandoned objects sometimes project shadows or reflections on the other elements of the scene. Strictly speaking, since such shadows and reflections are not present in the reference video, the algorithm tends to detect them, as seen in Fig. 6.2. However, the VDAO database does not consider them as objects of interest, which affects negatively the false positive measure.

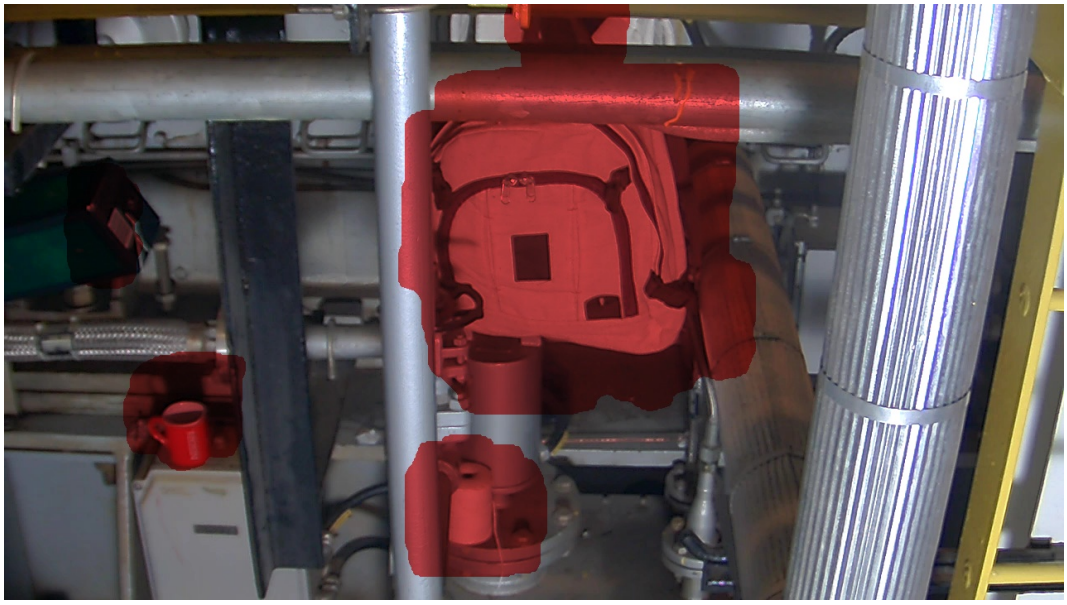
Based on all these facts, one should not expect ideal values of any pixel-based measurements, and an additional subjective validation scheme must also be employed to assess the overall system performance. However, in this parameter tuning procedure, we deemed the use of this pixel-based measurement an useful tool to, in a certain way, measure the percentage of the abandoned object covered by a detection spot, as this would help us better tune the system considering that it deals with abandoned objects of different sizes. This allows us to perform a finer parameter tuning than if we employed an object-level measurement in this procedure.

As seen in Chapter 3, the robot covers the 6 m track in about 3 minutes. In our experiments, we performed a temporal subsampling of 16 (1.5 frames/s) in order to try to cope with our real-time constraint.

Each parameter study employs, for the other variables, the values already obtained in the previous studies.

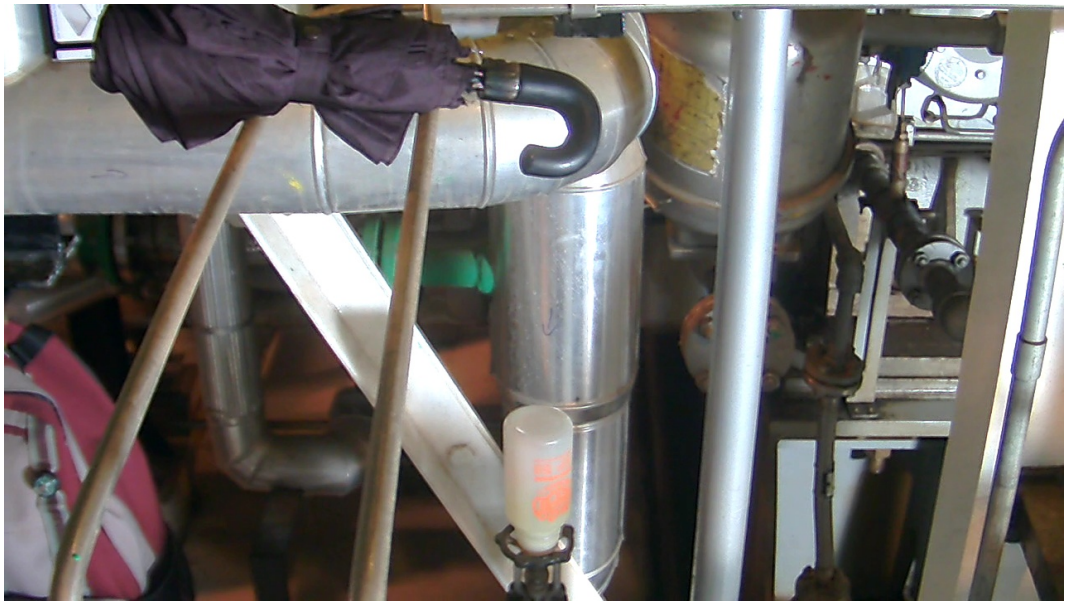


(a)

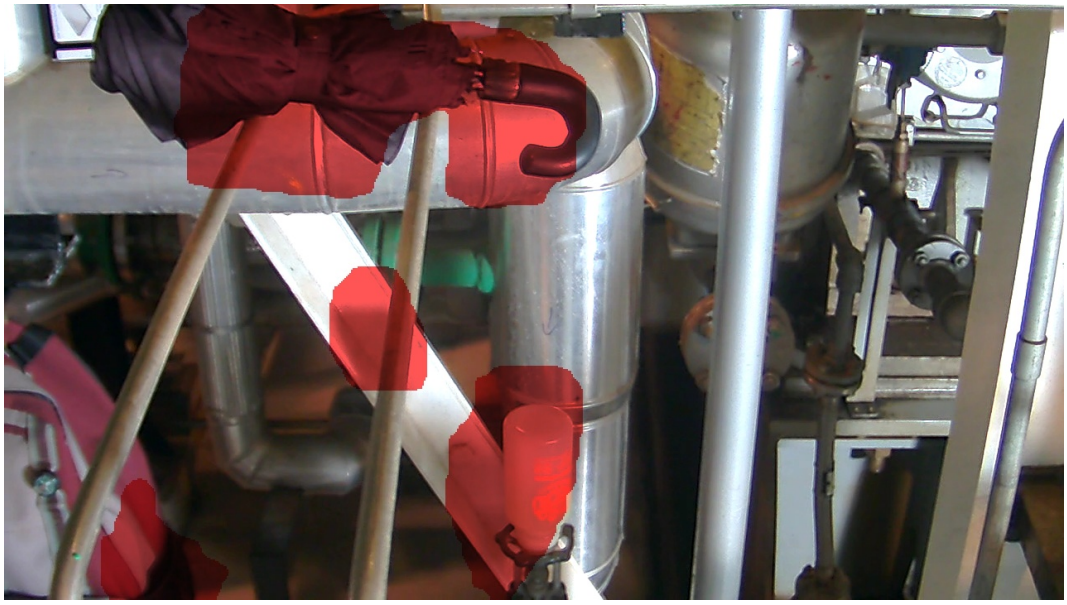


(b)

Figure 6.1: (a) The green box on the left is not being detected because it did not appear in enough frames yet; (b) Now that the green box has appeared in a sufficient number of frames, the algorithm started to detect it.



(a)



(b)

Figure 6.2: Example of shadow being detected as an abandoned object: (a) Target frame showing objects, and the shadow of an object (bottle); (b) Frame showing the object shadow also being detected.

6.1 NCC Binarization Threshold

The first important parameter in the proposed system is the threshold value b_t used to binarize the NCC function. Note that, from Eq. (5.2), its dynamic range is the interval $[-1, 1]$. In the discussion that follows, we assume that the NCC measure from Eq. (5.2) is normalized to be in the interval $[0, 255]$. A small value of b_t would generate too many false negatives, whereas large values of b_t would mark large numbers of abandoned-object as candidates to be processed by the system's subsequent stages. Fig. 6.3 shows the performance analysis curve for values of b_t in the set $\{60, 100, 140, 160, 190, 220, 250\}$. The smallest b_t corresponds to the lower leftmost point and the largest b_t to the upper rightmost point. From it, we can see that b_t sets a trade-off between true positives and false positives. Therefore, by analyzing the system's performance on a training set, an operator can control this trade-off by selecting a proper value of b_t. The performance analysis curve in Fig. 6.3 shows that for the training set, $b_t = 190$ provides a good trade-off, with a false positive rate of 1.4% and a true positive rate of 53%, while Table 6.1 shows the results of the performance analysis curve presented in Fig. 6.3 in details. Fig. 6.4 illustrates the effect of the NCC Binarization Threshold variable in our abandoned object detection process, and the pseudocode 1 shows in some detail the NCC step.

For this analysis, L_{tf} was set to 5, V was set to 20 and v_t was set to 10.

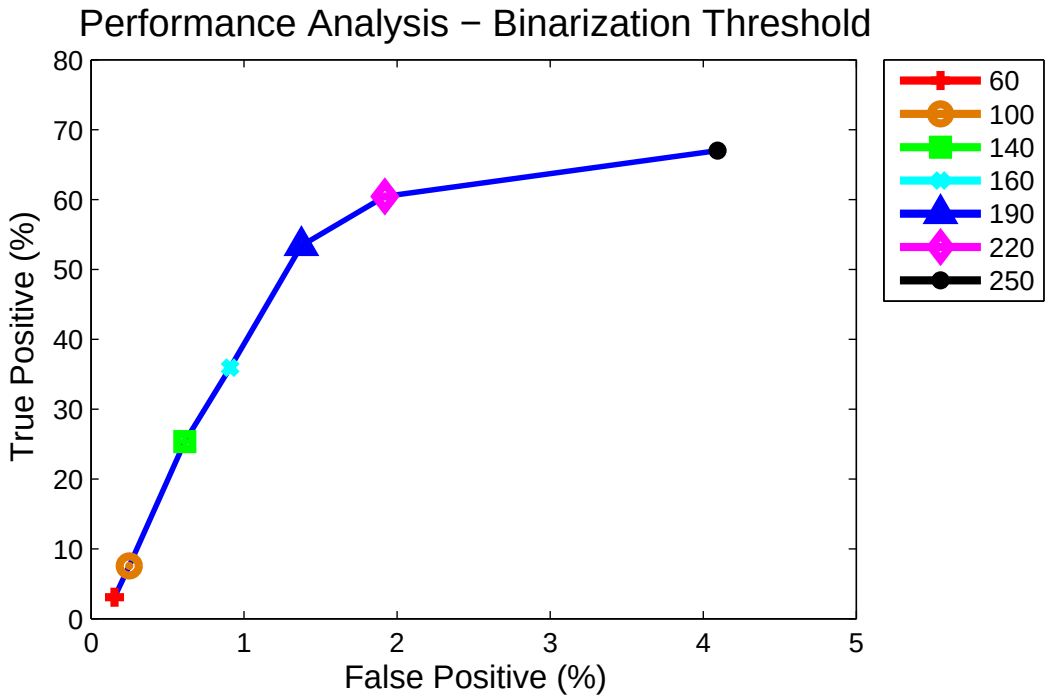
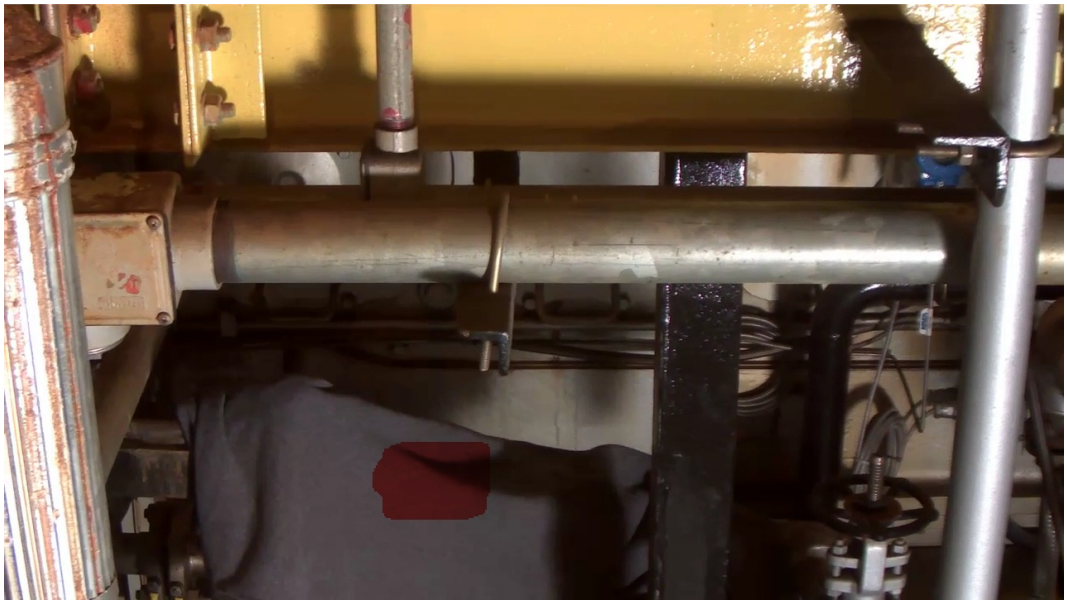


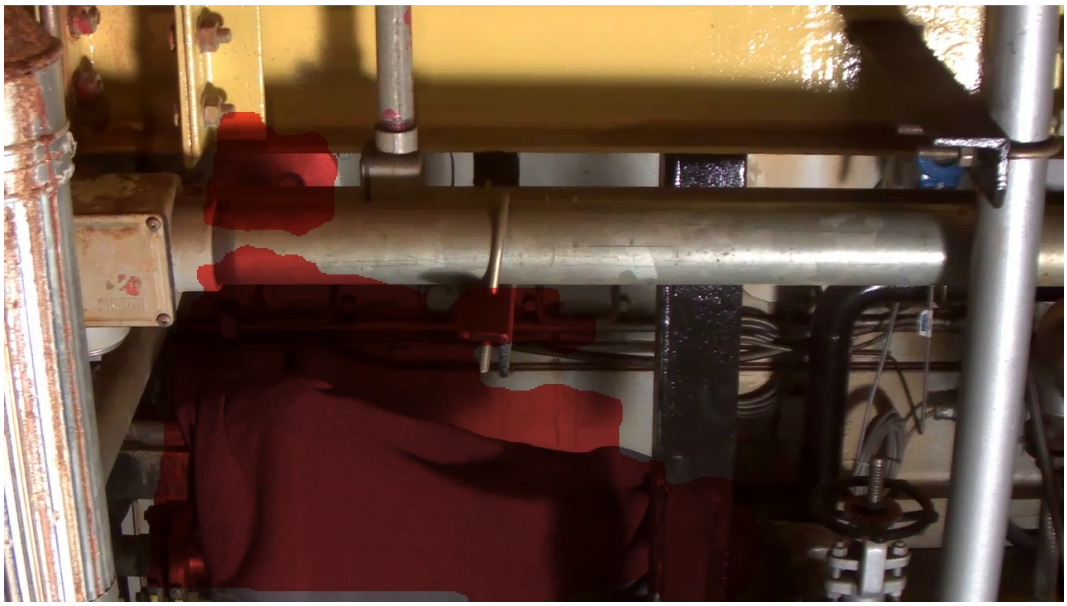
Figure 6.3: performance analysis curve for the NCC binarization threshold variable $b_t \in \{60, 100, 140, 160, 190, 220, 250\}$.

Table 6.1: Table showing the values obtained for the performance analysis curve relative to the Binarization Threshold variable.

Binarization Threshold (b_t)	True Positive μ (%)	True Positive σ (%)	False Positive μ (%)	False Positive σ (%)
60	3.06	6.66	0.15	0.44
100	7.56	10.74	0.25	0.62
140	25.35	20.43	0.61	0.91
160	35.94	24.13	0.91	0.98
190	53.39	20.07	1.37	1.04
220	60.45	19.15	1.92	1.34
250	67.00	18.24	4.09	2.40



(a)



(b)

Figure 6.4: Example of the impact of the b_t variable in our abandoned object detection process: (a) Detection with $b_t = 60$; (b) Detection with $b_t = 250$. The larger the value, the less restrictive the detection. Here one can even see the occurrence of false positives.

Algorithm 1 NCC step

Require: $TargetFrame$ and $ReferenceFrame$ {Geometrically Aligned}
  $displacement \leftarrow (K-1)/2$ {The NCC window has a $K \times K$ dimension}
  for $FrameRowCounter \leftarrow 1$ to total number of rows in frame **do**
    for $FrameColumnCounter \leftarrow 1$ to total number of columns in frame **do**
      $imageRowNumber \leftarrow FrameRowCounter - displacement$
      $imageColumnNumber \leftarrow FrameColumnCounter - displacement$
      for $NCCwindowRowCounter \leftarrow K$ **do**
        for $NCCwindowColumnCounter \leftarrow K$ **do**
          $NCCwindowReference(NCCwindowRowCounter, NCCwindowColumnCounter) \leftarrow ReferenceFrame(imageRowNumber, imageColumnNumber)$
          $NCCwindowTarget(NCCwindowRowCounter, NCCwindowColumnCounter) \leftarrow TargetFrame(imageRowNumber, imageColumnNumber)$
          $imageColumnNumber \leftarrow imageColumnNumber + 1$
        end for
        $imageColumnNumber \leftarrow FrameColumnCounter - displacement$
        $imageRowNumber \leftarrow imageRowNumber + 1$
      end for
      $NCCwindowReferenceMean(x,y) \leftarrow NCCwindowReference(x,y) - \bar{NCCwindowReference}$
      $NCCwindowTargetMean(x,y) \leftarrow NCCwindowTarget(x,y) - \bar{NCCwindowTarget}$
      $NCCimagePixel \leftarrow \left\langle \frac{NCCwindowReferenceMean}{\|NCCwindowReferenceMean\|}, \frac{NCCwindowTargetMean}{\|NCCwindowTargetMean\|} \right\rangle$
      {$\langle .,. \rangle$ is the inner product and $\|.\|$ is the L^2 norm}
      $NCCimage(FrameRowCounter, FrameColumnCounter) \leftarrow NCCimagePixel$
    end for
  end for
  for $FrameRowCounter \leftarrow 1$ to total number of rows in frame **do**
    for $FrameColumnCounter \leftarrow 1$ to total number of columns in frame **do**
      if $NCCimage(FrameRowCounter, FrameColumnCounter) \leq b_t$ **then**
        $BinarizedNCCimage \leftarrow 1$
      end if
    end for
  end for
  return $BinarizedNCCimage$ {This binary matrix contains possible abandoned object candidates}

6.2 Temporal Filtering Length

The second parameter of interest is the size of the temporal filter vector L_{tf}. As mentioned before, the temporal filtering is a necessary step to remove spurious false positives from the subsequent voting stage. Larger values of L_{tf} correspond to more restrictive temporal filters which may introduce false negative detections.

Fig. 6.5 shows the performance analysis curve associated to temporal filter vector size. From this curve, we can see that the best trade-off between true positives and false positives is given by $L_{\text{tf}} = 5$. It is indeed small enough to avoid most false negatives and large enough to deal with most false positives such as the one illustrated in Fig. 5.22, while Table 6.2 shows the results of the performance analysis curve presented in Fig. 6.5 in details. Fig. 6.6 illustrates the effect of the Temporal Filter Length variable in our abandoned object detection process. The bigger the vector, the smaller the resulting detection mask. The pseudocode 2 shows in some detail the Temporal Filtering step.

For this analysis, V was set to 20 and v_t was set to 10.

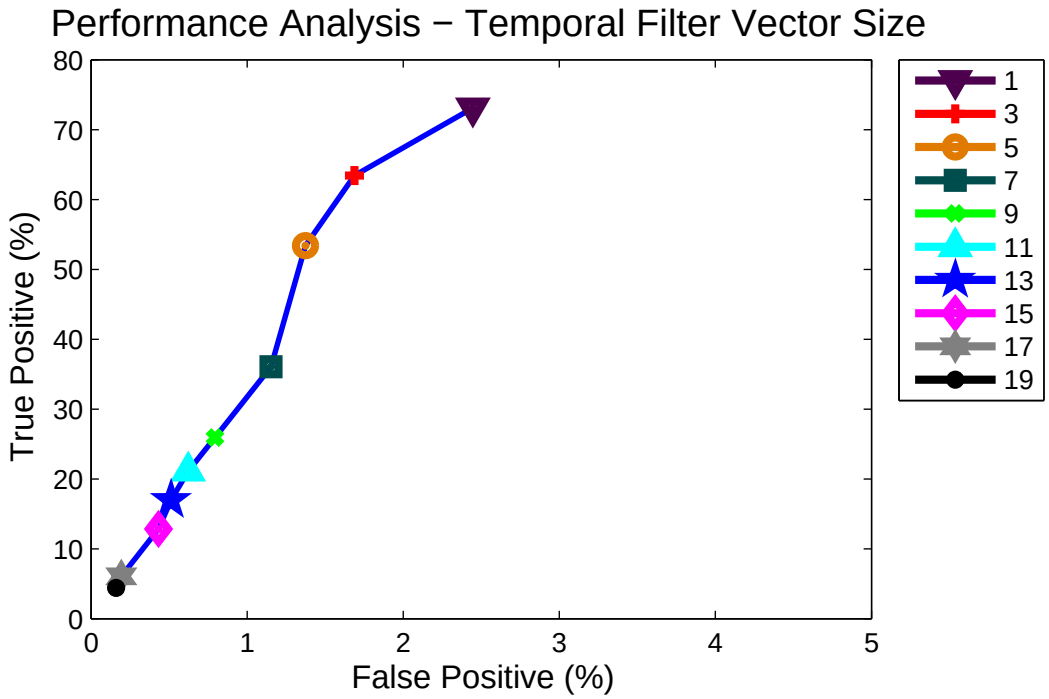
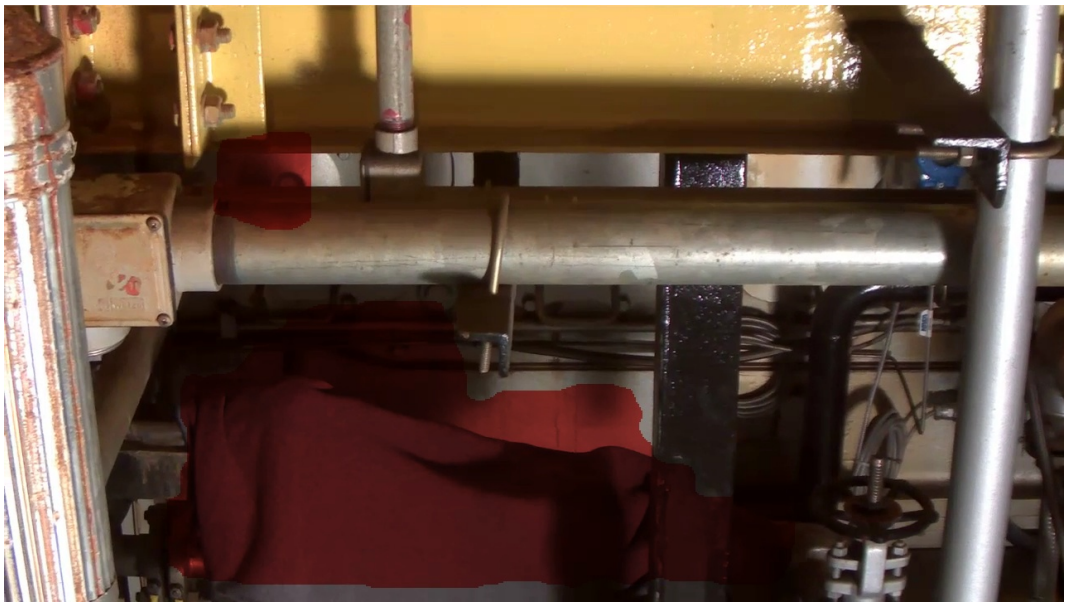


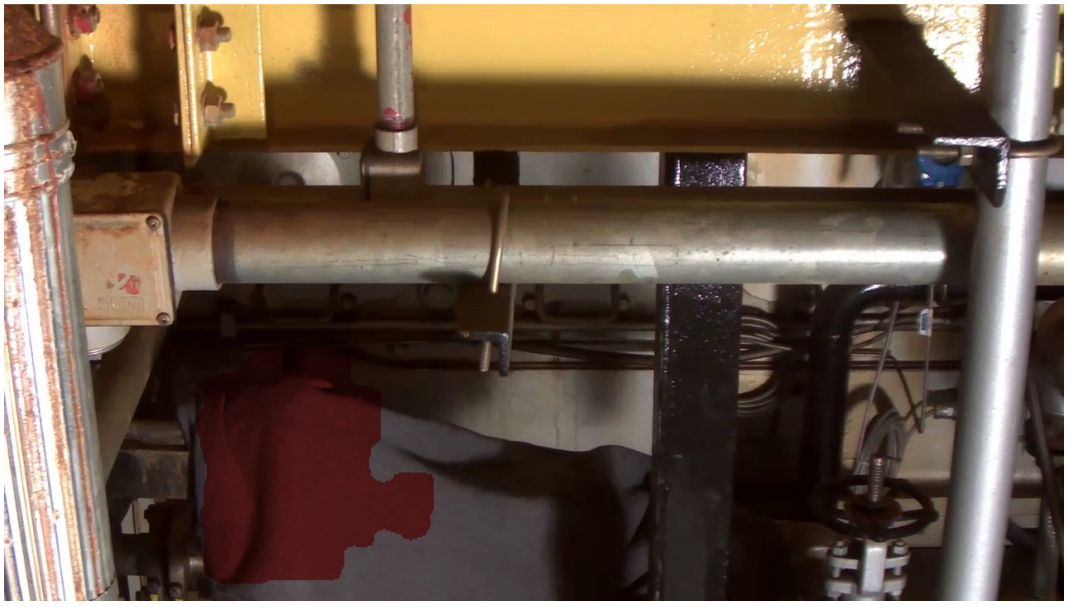
Figure 6.5: performance analysis curve for the temporal filter length $L_{\text{tf}} \in \{3, 5, 7, 9, 11, 13, 15, 17, 19\}$.

Table 6.2: Table showing the values obtained for the performance analysis curve relative to the Temporal Filter Length variable.

Temporal Filter Length (L_{tf})	True Positive μ (%)	True Positive σ (%)	False Positive μ (%)	False Positive σ (%)
1	73.11	18.38	2.45	1.77
3	63.43	19.97	1.69	1.21
5	53.39	20.07	1.37	1.04
7	36.04	27.19	1.15	1.05
9	25.92	27.22	0.79	1.09
11	21.12	22.96	0.62	0.90
13	16.96	18.43	0.51	0.75
15	12.83	14.57	0.43	0.62
17	6.07	11.09	0.19	0.36
19	4.39	10.37	0.16	0.37



(a)



(b)

Figure 6.6: Example of the impact of the L_{tf} variable in our abandoned object detection process: (a) Detection with $L_{tf} = 1$, where some false positive occurrences can be seen; (b) Detection with $L_{tf} = 19$. The larger the value, the more restrictive the detection.

Algorithm 2 Temporal Filtering step

Require: $NCCresult_{buffer}$ {buffer containing the last L_{tf} results of the NCC step} and $H1_{buffer}$ {buffer containing the homography matrixes computed for any two neighboring frames relatives to the results of the NCC step stored in $NCCresult_{buffer}$}

```
for u ← 1 to L_tf do
    M_1 ← NCCresult_buffer(u)
    for w ← u to L_tf − 1 do
        M_2 ← H1_buffer(w) × M_1
        M_1 ← M_2
    end for
    M_intersec ← M_intersec ∩ M_1
end for
return M_intersec {This binary matrix contains possible abandoned object candidates}
```

6.3 Voting Parameters

The pixel-level voting procedure on the detection mask depends on two parameters, namely the length in frames V of the voting interval and the threshold value v_t, as given in Eq. (5.4). When testing the influence of V, v_t was set to half the value of V, and the performance analysis curve in Fig. 6.7 has been obtained for V in the set $\{2, 6, 10, 16, 20, 24, 30, 40, 50, 60, 70, 80, 90\}$. From this curve, for $V \geq 50$, the true positive rates decrease without any improvement upon the false positive rates. This result is to be expected because, as the sizes of V and v_t increase, the number of objects that would not appear in a sufficient number of frames in order for them to be detected would also increase.

The best trade-offs are delivered with $V = 16$, which are then considered in a subsequent analysis on the value of v_t.

Table 6.3 shows the results of the performance analysis curve presented in Fig. 6.7 in details.

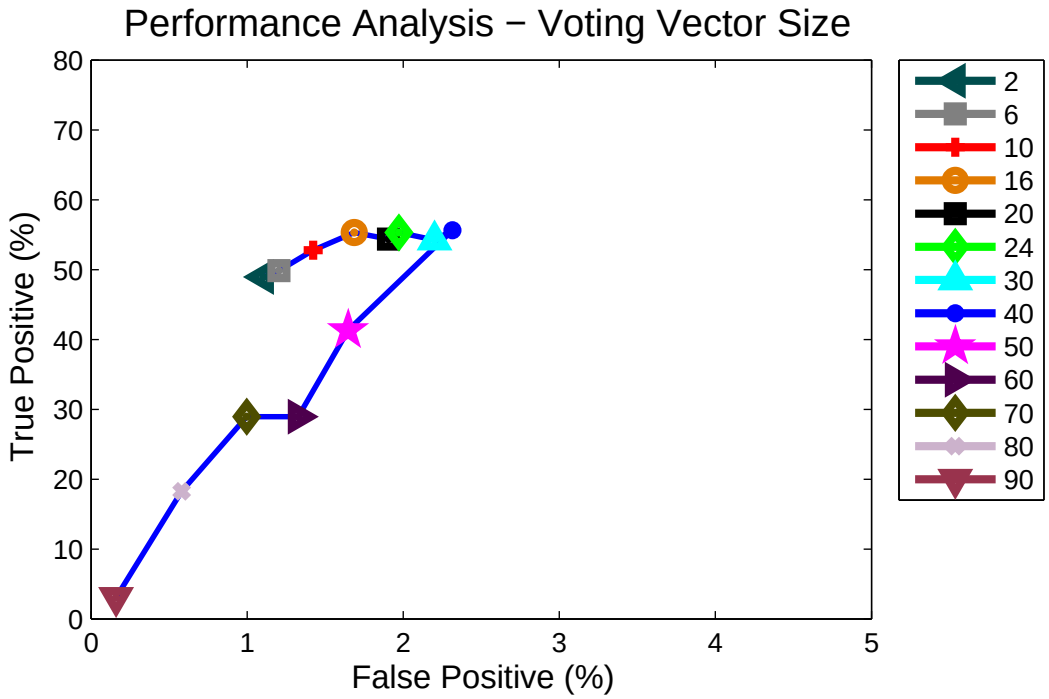


Figure 6.7: performance analysis curve for the Length in frames V of the voting interval $V \in \{2, 6, 10, 16, 20, 24, 30, 40, 50, 60, 70, 80, 90\}$.

Table 6.3: Table showing the values obtained for the performance analysis curve relative to the Voting Vector Length variable.

Voting Vector Length (V)	True Positive μ (%)	True Positive σ (%)	False Positive μ (%)	False Positive σ (%)
2	48.95	16.02	1.10	0.97
6	49.84	17.27	1.20	1.02
10	52.77	19.00	1.42	1.12
16	55.31	19.70	1.69	1.21
20	54.36	22.17	1.91	1.33
24	55.32	22.56	1.97	1.24
30	54.23	25.69	2.20	1.28
40	55.63	29.03	2.32	1.25
50	41.31	32.57	1.65	1.45
60	28.95	36.51	1.33	1.66
70	28.93	39.93	1.00	1.47
80	18.25	34.71	0.58	1.12
90	3.058	7.75	0.16	0.45

Fig. 6.8 illustrates the effect of the Voting Vector Length variable in our abandoned object detection process. The larger the vector, the smaller the resulting detection mask.

(a)

(b)

Figure 6.8: Example of the impact of the V variable in our abandoned object detection process: (a) Detection with $V = 2$, where some false positive occurrences can be seen; (b) Detection with $V = 50$. The larger the value, the more restrictive the detection.

Fig. 6.9 shows the results for $V = 16$ and $v_t \in \{1, 4, 7, 10, 13, 16\}$. From the performance analysis curves obtained from the study of the voting parameters, the best compromise is given by $V = 16$ and $v_t = 7$ frames, with a false positive rate of 1.83% and a true positive rate of 56.60%. Table 6.4 shows the results of the

performance analysis curve presented in Fig. 6.9 in details.

Fig. 6.10 illustrates the effect of the Voting Threshold variable in our abandoned object detection process, and the pseudocode 3 shows in detail the voting step.

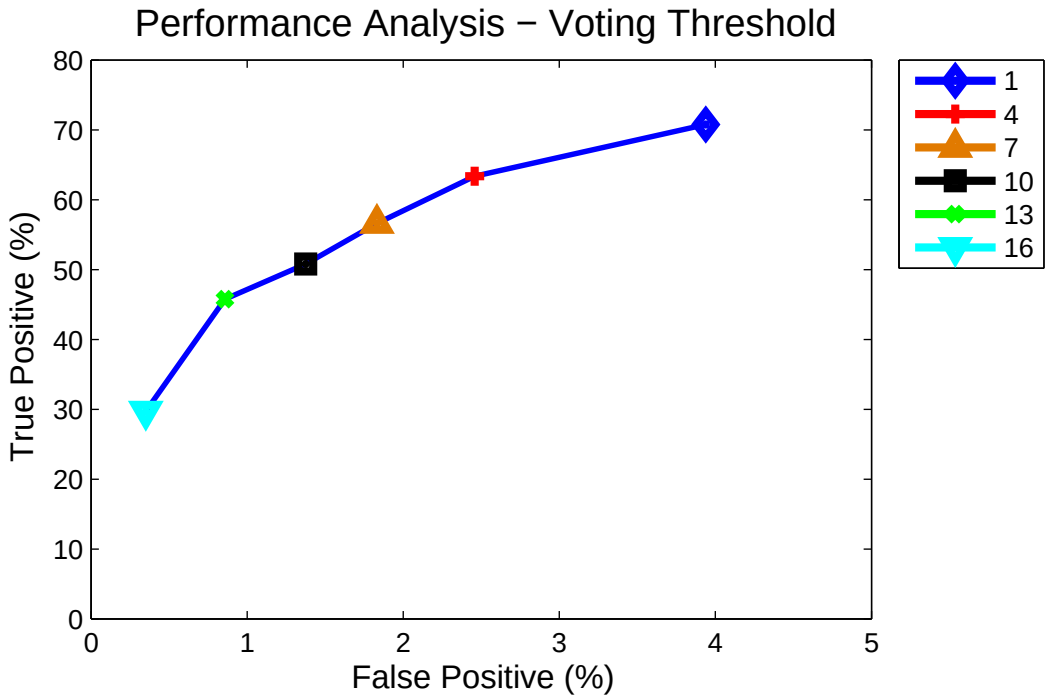
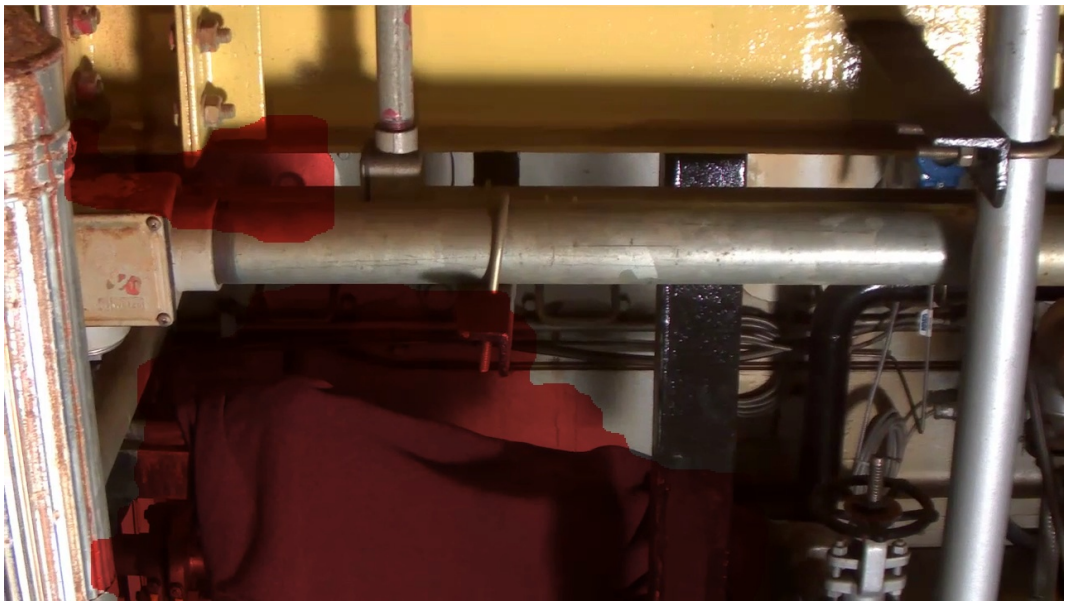


Figure 6.9: performance analysis curve for the threshold value $v_t \in \{1, 4, 7, 10, 13, 16\}$.

Table 6.4: Table showing the values obtained for the performance analysis curve relative to the Voting Threshold variable.

Voting Threshold (v_t)	True Positive μ (%)	True Positive σ (%)	False Positive μ (%)	False Positive σ (%)
1	70.76	17.87	3.94	2.70
4	63.35	20.47	2.46	1.72
7	56.60	20.72	1.83	1.30
10	50.78	19.80	1.38	1.05
13	45.77	16.47	0.86	0.85
16	29.73	19.61	0.35	0.54



(a)

(b)

Figure 6.10: Example of the impact of the v_t variable in our abandoned object detection process: (a) Detection with $v_t = 1$, where some false positive occurrences can be seen; (b) Detection with $v_t = 16$. The larger the value, the more restrictive the detection.

Algorithm 3 Voting step

Require: $TemporalFilterResult_{buffer}$ {buffer containing the last V results of the Temporal Filtering step} and $H2_{buffer}$ {buffer containing the homography matrixes computed for any two neighboring frames relatives to the results of the Temporal Filtering step stored in $TemporalFilterResult_{buffer}$}

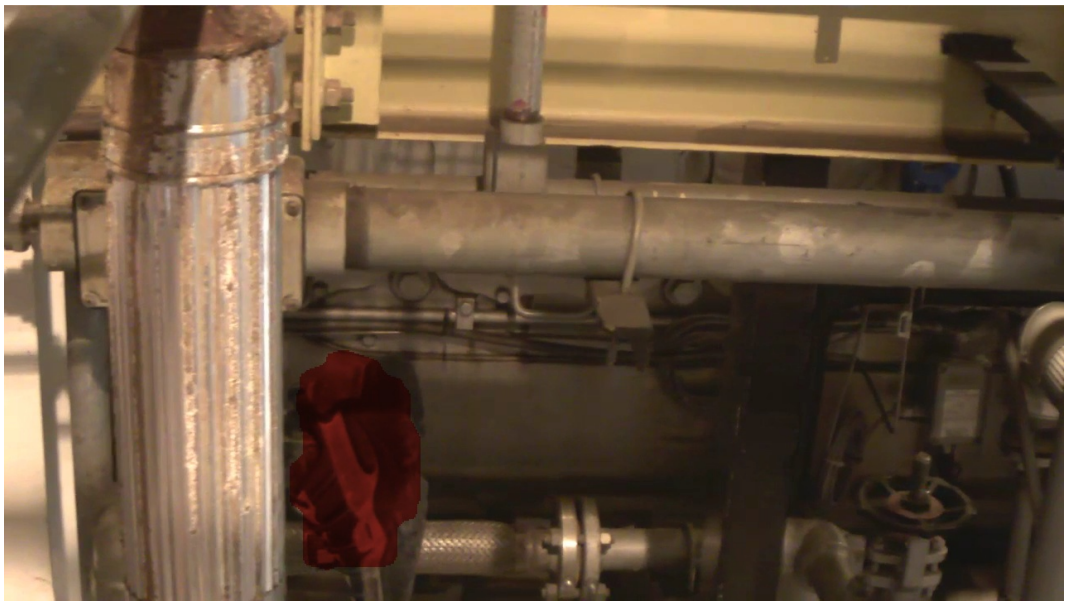
```
for u ← 1 to V do
    M1 ← TemporalFilterResult_buffer(u)
    for w ← u to V − 1 do
        M2 ← H2_buffer(w) × M1
        M1 ← M2
    end for
    for k ← 1 to CountMatrix_rows do {CountMatrix_rows is the number of rows
    in the matrix CountMatrix (the same number of rows in the video frames)}
        for l ← 1 to CountMatrix_cols do {CountMatrix_cols is the number of
        columns in the matrix CountMatrix (the same number of columns in the
        video frames)}
            if M1(k, l) = 1 then {If that element (pixel) in M1 is a candidate of
            belonging to a possible abandoned object}
                CountMatrix(k, l) ← CountMatrix(k, l) + 1
            end if
        end for
    end for
end for
for kgets1 to CountMatrix_rows do
    for lgets1 to CountMatrix_cols do
        if CountMatrix(k, l) ≥ v_t then
            M_voting ← 1
        end if
    end for
end for
return M_voting {This binary matrix contains the abandoned object candidates}
```

When performing the experiments for $V = 16$, however, we verified that the same phenomenon described in Section 5.4 and illustrated in Fig. 5.22 happened: the displacement of the detection mask caused by a foreground pipe. It is noteworthy that, in this experiment, we were performing the temporal filtering step.

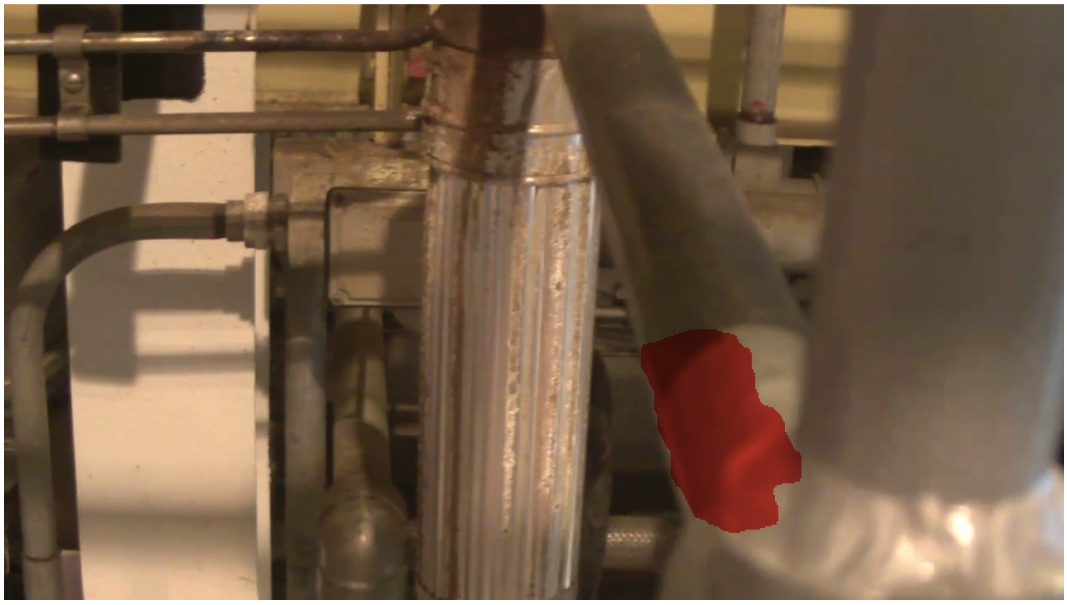
Fig. 6.11 shows the detection mask displacement phenomenon happening with $V = 16$ and $v_t = 7$.

We decided, then, to perform more tests varying L_{tf}, V (for $V \leq 16$ and for $V = 20$) and v_t for the shoe object to assess the situation. In the case of $V = 20$, we decided to perform this test because, as it can be seen in Table 6.3, the standard deviations of the true positive rates are large, in the order of 20%, allowing us to consider $V = 16$ and $V = 20$ as being roughly equivalents.

For $V = 20$, we were able to find a configuration in which we were able to deal with this effect, and still produce good detection results. The performance



(a)



(b)

Figure 6.11: Example of detection mask displacement phenomenon promoted by foreground pipe: (a) Detection mask placed over abandoned object, before the foreground pipe passes in front of the object; (b) Displacement of the detection mask happening after the foreground pipe passes in front of the abandoned object.

analysis curve relative to this experiment is shown in Fig. 6.12 for $V = 20$ and $v_t \in \{1, 4, 7, 10, 13, 16, 20\}$.

From this performance analysis curve, the best compromise is given by $V = 20$ and $v_t = 13$ frames, with a false positive rate of 1.37% and a true positive rate of 53.38%.

Table 6.5 shows the results of the performance analysis curve presented in Fig. 6.12 in details.

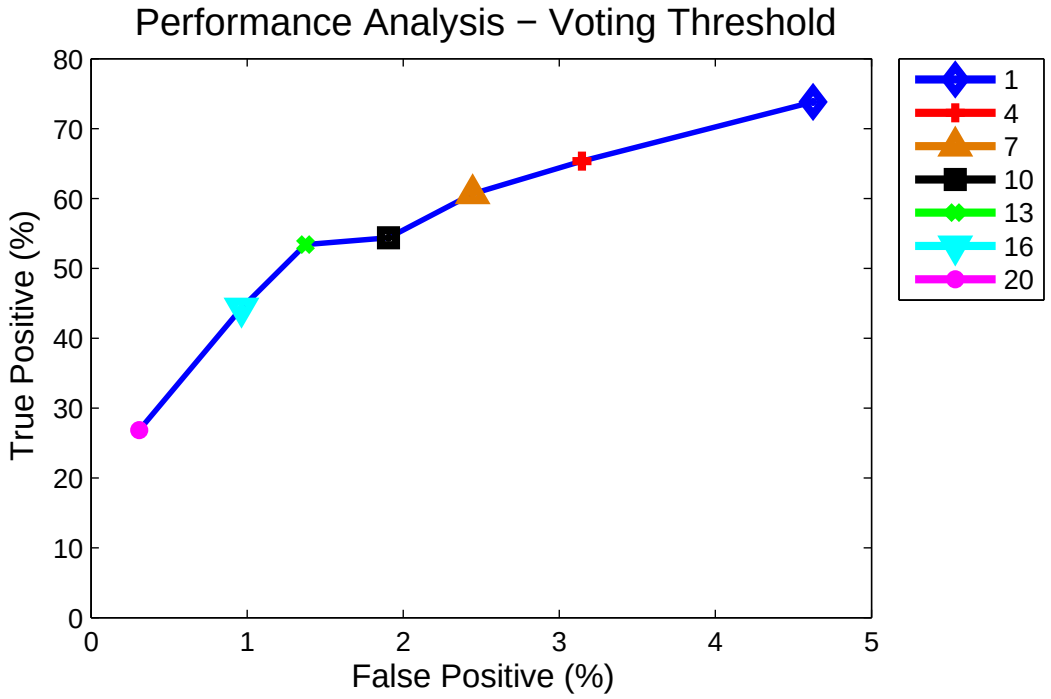
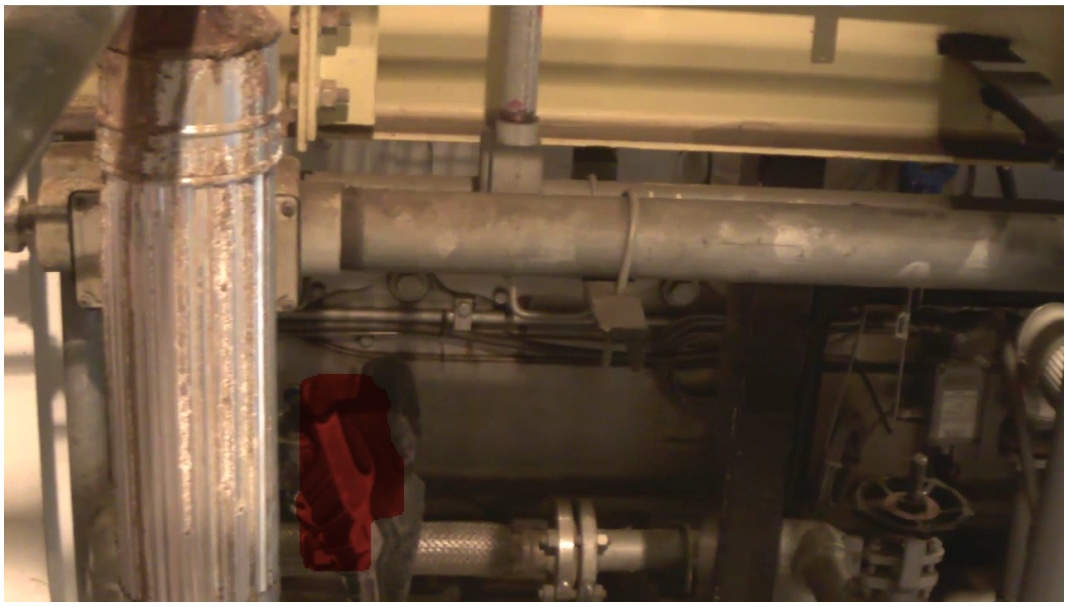


Figure 6.12: performance analysis curve for the threshold value $v_t \in \{1, 4, 7, 10, 13, 16, 20\}$.

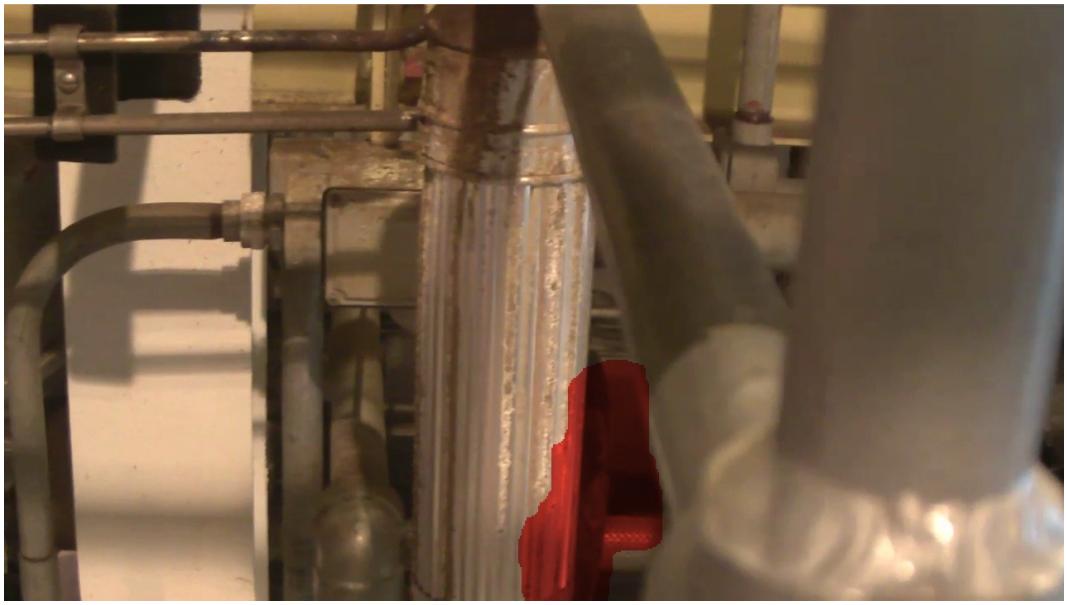
Table 6.5: Table showing the values obtained for the performance analysis curve relative to the Voting Threshold variable.

Voting Threshold (v_t)	True Positive μ (%)	True Positive σ (%)	False Positive μ (%)	False Positive σ (%)
1	73.80	17.21	4.63	2.93
4	65.34	20.65	3.14	2.01
7	60.66	21.84	2.44	1.56
10	54.36	22.17	1.91	1.33
13	53.39	20.07	1.37	1.04
16	44.31	19.10	0.96	0.88
20	26.82	21.37	0.31	0.50

Fig. 6.13 shows that the detection mask displacement phenomenon does not happen with $V = 20$ and $v_t = 13$.



(a)



(b)

Figure 6.13: With $V = 20$ and $v_t = 13$, the detection mask displacement phenomenon does not happen: (a) Detection mask is placed over abandoned object, before the foreground pipe passes in front of the object; (b) Detection mask is still properly placed even after the foreground pipe passes in front of the abandoned object.

For $V \leq 16$, by simply augmenting the value of L_{tf}, the detection mask displacement effect promoted by the foreground object (pipe) can be reduced, but not eliminated altogether. Also, larger values of L_{tf} can jeopardize detection, as seen in Section 6.2. However, by analyzing the situation considering the shoe videos with

and without extra illumination, we noticed that we were able to remove the detection mask displacement effect in the video with the spotlight illumination, but not in the video without this extra illumination. We also verified that, eventually, in the video without the spotlight illumination, more restrictive configurations led to results that were worse than the previous, less restrictive ones, with slightly bigger detection mask displacements.

Fig. 6.14 shows an example of detection mask displacement caused by the foreground object (pipe) also happening with a bigger Temporal Filtering Length ($L_{\text{tf}} = 7$) configuration (with $V = 16$ and $v_t = 7$).

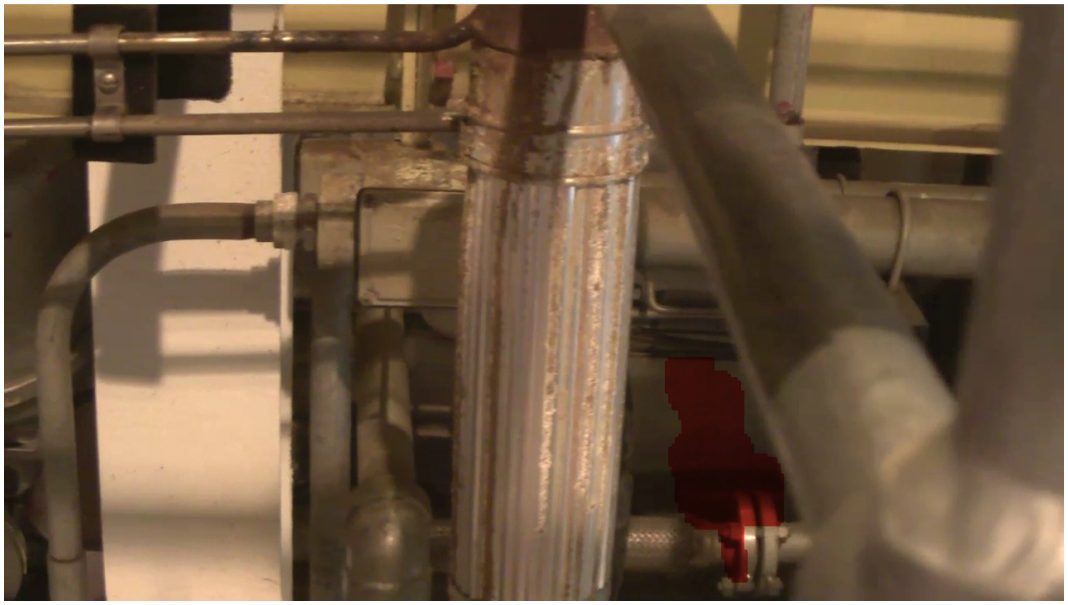


Figure 6.14: Example of detection mask displacement happening with a bigger Temporal Filtering Length ($L_{\text{tf}} = 7$) in video without extra illumination (with $V = 16$ and $v_t = 7$).

Fig. 6.15 shows the detection mask being properly placed in the video with the spotlight illumination, even after the foreground pipe passes in front of the abandoned object (shoe), with $V = 16$ and $v_t = 7$.

This situation led us to look into what was happening in the Homography Transformation step (described in Section 5.2). We verified that, in the video without the extra illumination, when the foreground object (pipe) appeared, it produced a small variation in the homography, amplifying the detection mask displacement effect. As this effect can be cumulative, it can explain why, eventually, more restrictive configurations led to worse results, as described above.

Fig. 6.16 shows the small variation in the Homography Transformation that happens when the foreground object (pipe) appears.

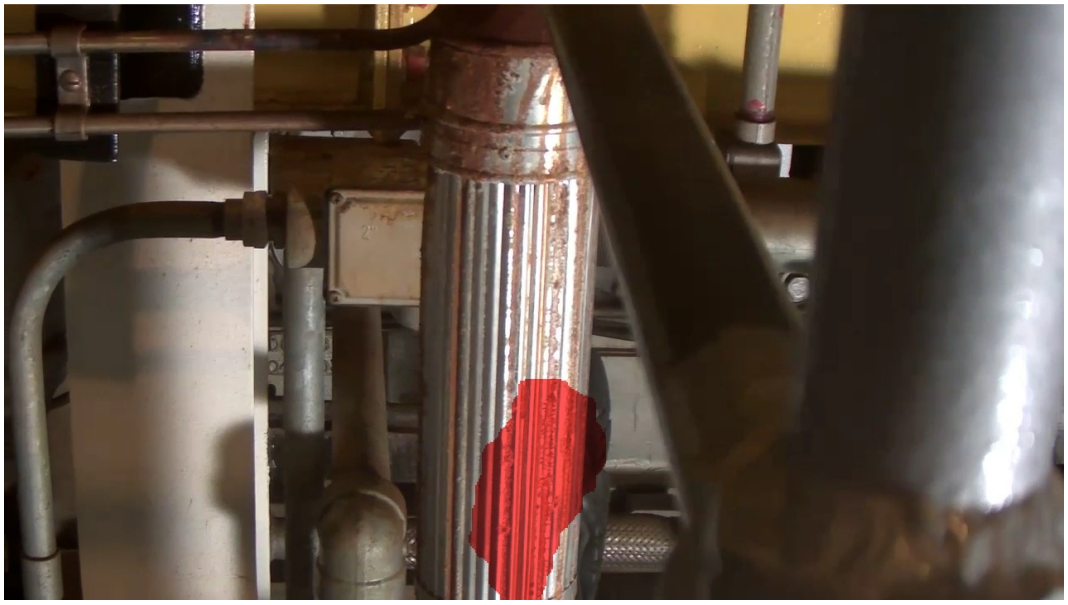
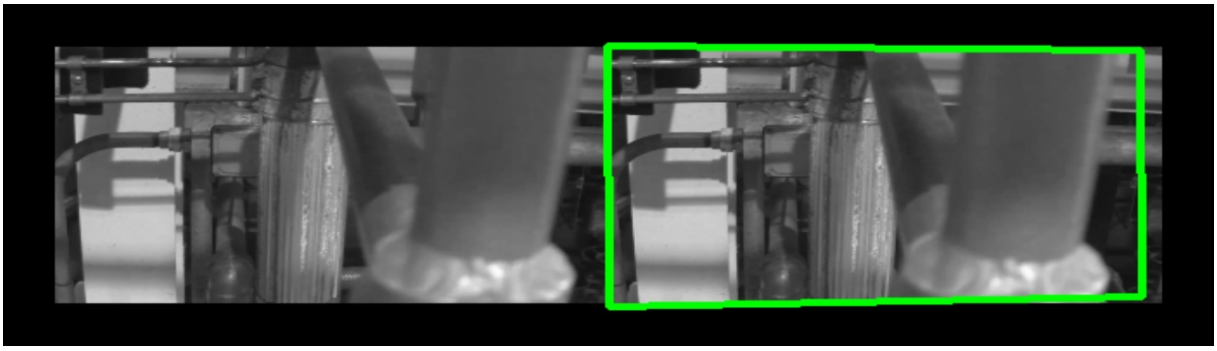
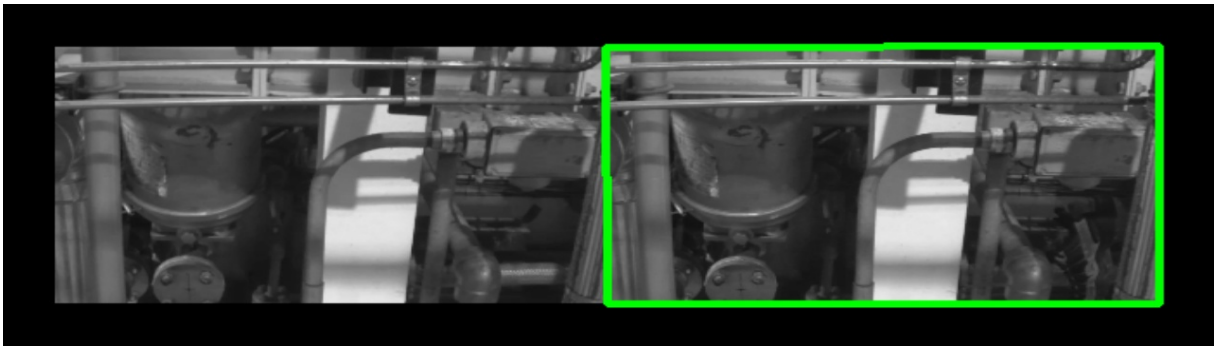


Figure 6.15: Example of detection mask being properly placed in the video with the spotlight illumination (with $V = 16$ and $v_t = 7$).



(a)



(b)

Figure 6.16: The green quadrilateral shows the image-plane mapping from the left image into the right one: (a) Homography Transformation varying when foreground object (pipe) appears; (b) Proper Homography Transformation being generated after foreground object (pipe) fades out.

As the $V = 16$ and $v_t = 7$ configuration was able to deliver an adequate trade-off between the true and false positive rates, and as the detection mask displacement phenomenon shown here was mostly produced by an issue happening with an Homography Transformation in our Image Registration step in a single video, we deemed this configuration as the most suitable to follow up on our experiments, as it was obtained with our proposed system parameters tuning procedure.

Table 6.6 sumarizes the values obtained for each variable studied in this chapter:

Table 6.6: Table showing the values obtained for the studied variables.

Variable Name	Variable	Value
NCC Binarization Threshold	b_t	190
Temporal Filtering Length	L_{tf}	5
Voting Vector Length	V	16
Voting Threshold	v_t	7

Chapter 7

Experimental Results

In this chapter, we assess the parameter tuning procedure presented in Chapter 6 and the overall system performance using the remaining database videos not employed in the parameter tuning stage.

In order to assess the parameter tuning procedure we compute the true positive and false positive rates (as defined in Chapter 6) of the videos from the validation set. The frame-based detection results (average (μ) and standard deviation (σ) values) are shown in Table 7.1 for both the validation and training sets. From these numbers, we can conclude that the parameter tuning detailed in Chapter 6 yields similar and consistent results with both the training and validation sets. It is important to note that despite the not so high values of true positive rates (of the order of 60%), these refer to the percentage of the bounding boxes covered, as specified in Chapter 6. However, in a practical application, an operator will see the masks and decide whether an abandoned object has been detected or not. We have performed this evaluation in the validation set, and verified that all the abandoned objects contained in the validation dataset were properly detected.

Table 7.1: Table showing the comparison between the detection results obtained with the training and validation sets.

Data	True Positive		False Positive	
Employed	μ (%)	σ (%)	μ (%)	σ (%)
Training set	57	21	1.8	1.3
Validation set	66	15	3.6	3.6

In order to assess the overall system performance, we have employed our testing set, consisting of the 3 multi-object videos from the VDAO database [54], with 15 abandoned objects in each video. This multi-object scenario requires more complex metrics than before, as in principle we do not know beforehand which ground-truth frame mask corresponds to the mask of a given detected object. In addition, in cases of missed or false detections, there is no one-to-one correspondence with the

ground truth masks. Due to these matters, our initial system evaluation was based on [65], which proposes metrics that take into account the accuracy of the matchings as well as the false positives and false negatives for the case of multiple objects. In addition, we also assess the proposed method using metrics that mimic whether or not an operator, by looking at the detection masks generated, is able to indicate correctly the presence of an abandoned object.

The definition of the metrics in [65] is based on two pixel sets:

- $A_{k,i}$, the set of pixel positions belonging to the ground truth mask of the object of index i in frame k.
- $\hat{A}_{k,i}$, the set of pixel positions belonging to the detected mask of the object of index i in frame k.

The first metric is the accuracy error $\mathcal{A}_k$. It represents the amount of spatial overlap between the detected-object bounding box $\hat{A}_{k,i}$ and its corresponding ground-truth value. It has to be computed over all possible correspondences of masks from the detected objects and the ground truth masks. If we define

$$\mathcal{O}(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{7.1}$$

where $|A|$ is the number of elements of the set A, then the accuracy error is given by

$$\mathcal{A}_k = \min_{\pi \in \Pi_{\max(d_k, \hat{d}_k)}} \sum_{i=1}^{\min(d_k, \hat{d}_k)} [1 - \mathcal{O}(\hat{A}_{k,i}, A_{k,\pi(i)})], \tag{7.2}$$

where d_k and $\hat{d}_k$ are respectively the number of ground truth masks and detected objects in frame k, the permutation $\pi(i)$ is a one-to-one function that maps the set of indexes of detected objects into the indexes of ground truth masks, and Π_j is the set of all possible permutations over j indexes.

The frame-level accuracy error rate (AER) is defined as the average of $\mathcal{A}_k$ over all frames, that is

$$\text{AER} = \frac{1}{K} \sum_{k=1}^{K} \mathcal{A}_k, \tag{7.3}$$

where K is the total number of frames. The minimum accuracy error happens when all the bounding boxes coincide exactly pixel by pixel, and is therefore zero. The maximum value happens when no bounding boxes coincide, and is equal to $\min(d_k, \hat{d}_k)$.

The problem with $\mathcal{A}_k$ is that if an object is missed or there is a false positive, its contribution to $\mathcal{A}_k$ is zero. To cope with that issue, the authors of [65] define the cardinality error rate, that quantifies the discrepancy in estimating the number of

targets. It is just the average difference in the numbers of ground truth masks and detected objects over all frames, that is

$$\mathrm{CER} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{C}_k. \tag{7.4}$$

where $\mathcal{C}_k = |d_k - \hat{d}_k|$.

Therefore, a metric that would take into account both the accuracy and the effect of false positives and false negatives can be obtained by adding both $\mathcal{A}_k$ and $\mathcal{C}_k$. In addition, since both increase with the number of objects present, this sum can be normalized by the number of objects to produce the METE_k (Multiple Extended-target Tracking Error) score, defined as [65]

$$\mathrm{METE}_k = \frac{\mathcal{A}_k + \mathcal{C}_k}{\max(d_k, \hat{d}_k)} \tag{7.5}$$

In [65] it is shown that the METE value is equal to 0 in the case of perfect matching and is 1 in the worst case.

From all the metrics to evaluate tracking that are presented in [65], we chose the one presented above (METE) because it can also be applied in our anomally detection scenario, as it represents the extent of the mismatch between the actual abandoned object and the detected one in a given frame, combined with a cardinality error.

Table 7.2 shows the AER, CER and METE results, averaged over all frames, using our testing video set, with a temporal subsample value of 16, as described in Chapter 6.

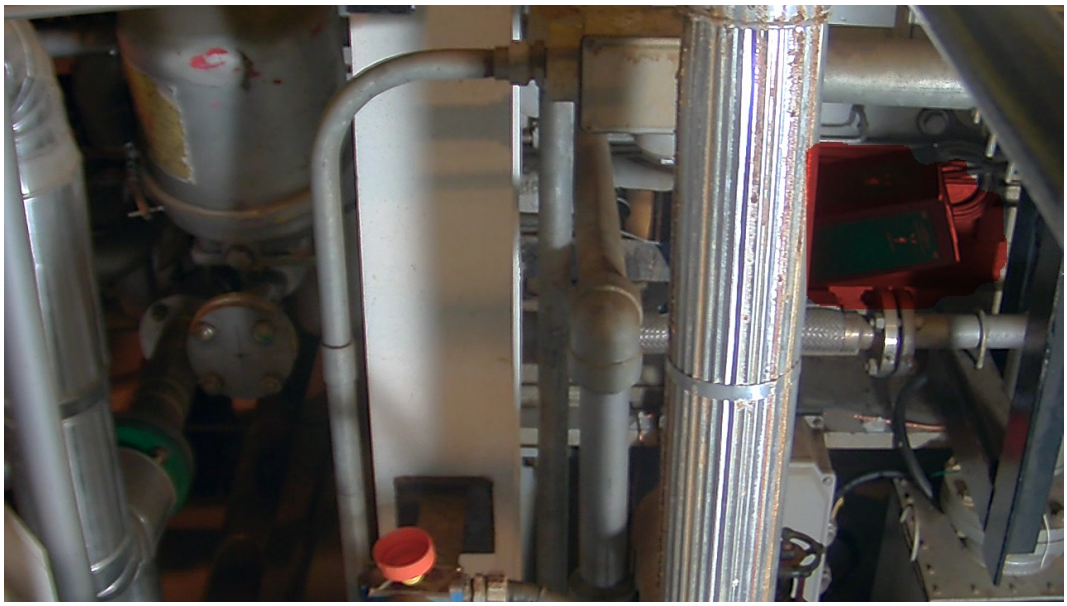
Table 7.2: Quantitative evaluation of object-detection system performance using the METE metrics with temporal subsampling of 16.

Data	AER	CER	METE	
Employed	μ	μ	μ	σ
Multiobject video 1	2.1	1.6	0.80	0.10
Multiobject video 2	1.8	1.2	0.72	0.17
Multiobject video 3	2.5	1.2	0.80	0.10

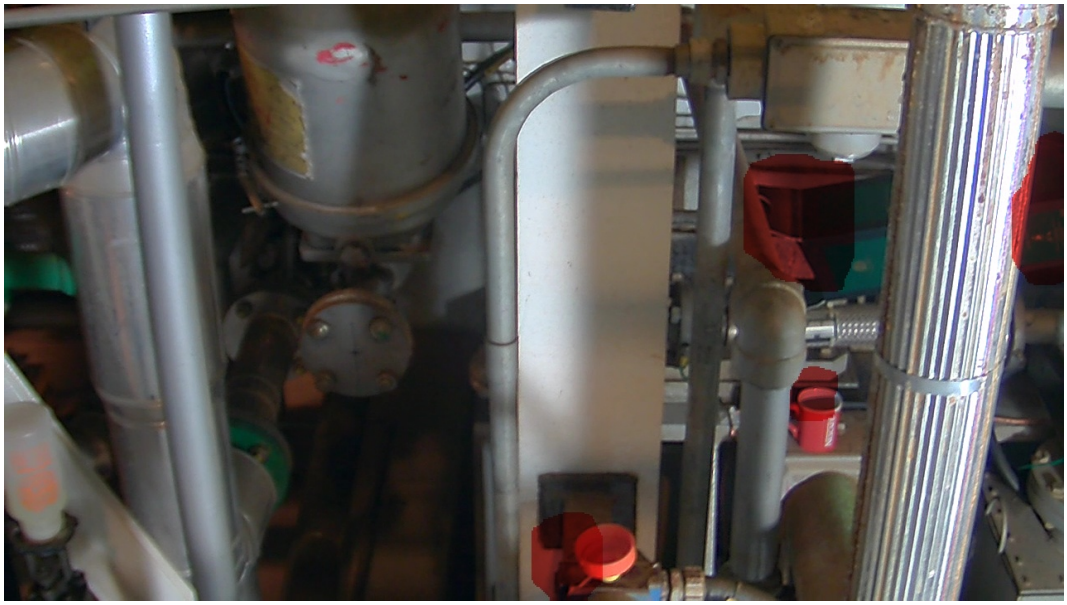
When performing a visual validation of the system performance, we noticed that it missed 2 objects in the first video, 2 in the second, and 1 in the third (each video has 15 objects). In all cases, the missed objects were slightly smaller than the ones considered in the system development. Hence, the resulting detection mask was too small, and some parallax issues caused by the homography transformation forced the temporal filter to remove these masks. In addition, due to the object position, one of these missed objects also appeared in a very limited number of frames, making

its detection even harder. When reducing the frame decimation factor by 2, which is equivalent to reducing the robot speed in half, all problems were eliminated and all objects were properly detected.

Fig. 7.1 shows an example of previously undetected object (bottle cap) being detected with reduced frame decimation.



(a)



(b)

Figure 7.1: Example of false negative: (a) Frame showing missed detection (bottle cap); (b) Frame showing bottle cap detected, with reduced frame decimation. One can notice the detection spot displacement over it due to parallax effect.

Thus, the following evaluations were performed with detection videos generated with a temporal subsample of 8, but with the same parameter configurations as

before.

Table 7.3 shows the AER, CER and METE results, averaged over all frames, using our testing video set, with a temporal subsample value of 8.

Table 7.3: Quantitative evaluation of object-detection system's performance using the METE metrics with temporal subsampling of 8.

Data	AER	CER	METE	
Employed	μ	μ	μ	σ
Multiobject video 1	2.5	0.9	0.72	0.11
Multiobject video 2	2.5	2.2	0.77	0.14
Multiobject video 3	3.2	2.9	0.85	0.09

The high METE scores, around 78% for each video, indicate significant discrepancies between the bounding boxes of the detected objects and the ground truth, what is easily understood, as discussed in Chapter 6, by the situation depicted in Fig. 7.2. In such a case, only about 50% of the bounding box for the string roll detection mask, in the middle bottom of the frame, coincides with its ground truth, justifying the resulting large METE values.

Figure 7.2: Detection results for frame with objects of different sizes (backpack on the left, box on the right and string roll in the middle). All of them have been correctly detected.

Despite the large METE values yielded by the proposed system, an operator can correctly indicate the presence of the string roll by looking at the provided detection mask. The same is true for the backpack and the box objects, also shown in that frame image, and the object-based detection error rate should be zero in that case. This demonstrates the importance of measuring the detection error rate at the object level, which can be done based on the following measurements:

- The number of true positives N_{TP_j}, that is the number of frames where object j is correctly detected.
- The overall number of false positives $N_{\mathrm{FP}_{\mathrm{all}}}$, that is the number of frames where any object is incorrectly indicated.
- The number of false negatives N_{FN_j}, that is the number of frames where object j is incorrectly missed.
- The overall number of true negatives $N_{\mathrm{TN}_{\mathrm{all}}}$, that is the number of frames that have been correctly indicated as having no objects.
- The number of positives N_{P_j}, that is the number of frames where object j is present.
- The overall number of positives N_{P}, that is the number of frames where the presence of any object is indicated.
- The number of negatives N_{N_j}, that is the number of frames where object j is not present.
- The overall number of negatives N_{N}, that is the number of frames where no object is indicated.

Note that the number of true negatives cannot be computed for a particular object j, since the frames that have been correctly classified as having no object cannot be associated with any object. In addition, one cannot assign a false positive to any object. Then, using the above measurements the following detection error rate metrics are defined:

$$\mathrm{TP} = \frac{\sum_{j=1}^{\mathrm{N_{objs}}} N_{\mathrm{TP}_j}}{\sum_{j=1}^{\mathrm{N_{objs}}} N_{\mathrm{P}_j}} \tag{7.6}$$

$$\mathrm{FP} = \frac{N_{\mathrm{FP}_{\mathrm{all}}}}{N_{\mathrm{P}}} \tag{7.7}$$

$$\mathrm{TN} = \frac{N_{\mathrm{TN}_{\mathrm{all}}}}{N_{\mathrm{N}}} \tag{7.8}$$

$$\mathrm{FN} = \frac{\sum_{j=1}^{\mathrm{N_{objs}}} N_{\mathrm{FN}_j}}{\sum_{j=1}^{\mathrm{N_{objs}}} N_{\mathrm{N}_j}} \tag{7.9}$$

where $\mathrm{N_{objs}}$ is the total number of objects.

Table 7.4 shows the resulting values for these metrics using our multi-object testing set. The average number of true positives as defined by Eq. (7.6) is approximately 71%, and the average number of false positives (Eq. (7.7)) is about 42%, most of them due to the object shadows and reflections, which are multiplied in the

cluttered environment, as already discussed in Chapter 6. In addition, one can see that the average value of false negatives (Eq. (7.9)) is approximately 10%, meaning that there are not many missed frames containing abandoned objects. This indicates that the proposed abandoned object detection system can be useful for providing effective alarms of the presence of abandoned objects in a practical situation. Note that the void entries in Table 7.4 are accounted for the fact that, in the particular case of the multi-object videos of the VDAO databases, there are no frames without any abandoned objects. Thus, both $N_{\mathrm{TN_{all}}}$ and N_{N} are zero, and TN is undefined.

Table 7.4: Quantitative evaluation of object-detection system's performance in terms of the correct detection of an abandoned object (Eqs. (7.6) to (7.9)).

Data Employed	TP (%)	FP (%)	TN (%)	FN (%)
Multiobject video 1	68	40	—	13.4
Multiobject video 2	69	42	—	9.6
Multiobject video 3	77	43	—	7.7

The proposed metrics given in Eqs. (7.6)–(7.9) are relevant for the assessment of the overall system capabilities and its intrinsic behaviors. In a practical surveillance scenario, however, one is mostly interested in a system that can give an alarm for all abandoned objects. If the alarm happens in all frames that the object is present or in just some, the practical effect is the same, drawing the attention of an operator that can further analyze the surveillance video. We have performed this analysis on the three videos from the testing set, and have verified that the proposed system has alarmed all 15 objects in the 3 videos. This suggests the usefulness of the proposed method in a practical surveillance scenario.

One may argue that the results shown are incomplete, since they lack a comparison with the state-of-the-art in abandoned object detection using moving cameras. An interesting work about detection of abandoned objects using a moving camera is the one in [8]. However, that method has been developed for the detection of abandoned objects in roads, that have several particular characteristics, such as: (i) the presence of horizon lines, (ii) objects that are far enough from the camera such that the effects of parallax have very little influence on the results, and (iii) the use of a GPS to synchronize the videos. Thus, the assessment of this method using the VDAO database would not be fair, since this database does not satisfy any of those conditions. Also, the databases used in most of the other works on abandoned object detection using moving cameras are not public, which is another obstacle to a fair comparison. Nevertheless, we expect that, since we made the VDAO database public, it will be easier from now on for other authors to compare their results on the VDAO database.

As far as complexity is concerned, for real-time applications we have to be able

to run the detection algorithm in the time interval between two frames. If this time interval is not large enough due to processing power restrictions, we have two options:

- To subsample the videos. The problem with that situation is that if a video is subsampled by a very large factor, there may not be enough frames where an object is present in order for us to perform the temporal filtering and voting procedures (Eqs. (5.3) and (5.4)).

- To reduce the robot's speed and then subsample the videos. In this case there may be both enough time for processing between frames and an adequate number of frames for the temporal filtering and voting.

In our experiments, we employed an Intel core I7 2630QM processor with a 2-GHz clock rate, and with 8 GB of RAM. With this configuration, considering that the algorithms used in our system have not been optimized, each frame took about 4.5 s to be processed. Considering our tests temporal subsampling, each frame should take about 0.7 s to be processed for us to consider that the system operates in real time. By decomposing the 4.5 s total time in each of our algorithms times, we have:

- 0.05 s for the SURF with the 1° absolute angular displacement restriction ($\mu = 0.05$ s, $\sigma = 0.006$ s and worst case = 0.06 s). The SURF alone takes about 0.045 s. This is performed 3 times each iteration, one to register the reference and target frames, and two others to register the target frames used in the Temporal Filtering and Voting steps. This way, we have, here, 0.15 s;

- 0.015 s for the NCC step ($\mu = 0.015$ s, $\sigma = 0.003$ s and worst case = 0.017 s);

- 0.36 s for the Temporal Filtering step ($\mu = 0.36$ s, $\sigma = 0.02$ s and worst case = 0.37 s);

- 3.9 s for the Voting step ($\mu = 3.9$ s, $\sigma = 0.16$ s and worst case = 4.0 s);

- 0.014 s for the morphological operations ($\mu = 0.014$ s, $\sigma = 0.002$ s and worst case = 0.02 s).

However, if we employ the geometric registration proposed and tested in [59], we could reduce this time by a factor of 3. Also, if we precalculate PoIs and the homographies between the reference frames and use them instead of using the ones calculated using the target frames in the Temporal Filtering and the Voting steps, the PoIs would be calculated only once in each algorithm iteration, instead of thrice. This way, instead of 0.15 s, the registration step wold take 0.015 s.

In our Voting step, it is the many homographies among the many frames relatives to the detection masks stored in the Voting Vector that are time consuming (see the nested for loops in pseudocode 3). With $V = 16$, for example, we calculate 120 homographies (15 from the first element in the vector to the last, plus 14 from the second element to the last, and so on). But if we employ the strategy of precalculating the homography matrices using the reference frames, we would only have to apply them once in each detection mask present in the Voting vector. This way, we would perform 15 operations instead of 120. This could reduce the 3.9 s time to 0.5 s

The same reasoning can be considered in the Temporal Filtering step (see the nested for loops in pseudocode 2): with $L_{\text{tf}} = 5$, instead of 10 operations, we would have 4. This way, this step would take 0.15 s to be performed.

With these mentioned optimizations, the time to process each frame could go down to 0.7 s.

Also, all our operations are being performed sequentially. If we paralelize the operations that allow it, we could also improve our algorithms performance.

Finally, one could always employ a faster computer ir order to obtain faster results.

We should also take into account that we are employing a multiscale algorithm (in our tests, we employed 4 resolutions). If the operator decides to search for even smaller abandoned objects, more (and bigger) resolutions would be necessary, and our processing time would go up. Likewise, if he decides to search only for bigger objects, less resolutions would be needed (the bigger one(s) would not be used), and the resulting processing time would go down. Also, with a different number of resolutions, and as the NCC window dimensions could be different than the $K = 5$ employed in our tests, the process of tuning the system's parameters shown in Chapter 6 would have to be performed considering these new requirements (that is, finding even smaller objects, or not having to find small objects).

Figs. 7.3 and 7.4 show examples of successfully detected abandoned objects in videos containing single objects. Fig. 7.5 shows examples of objects successfully detected in videos containing multiple abandoned objects.

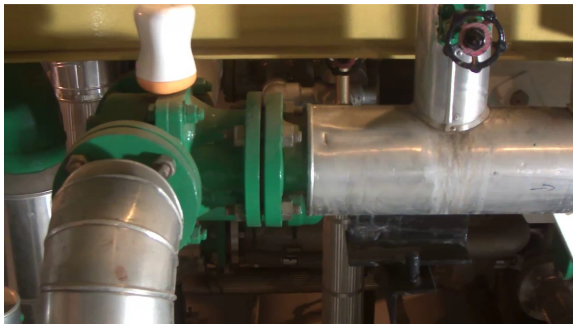
(a) Target image (camera box)

(b) Detected object (camera box)

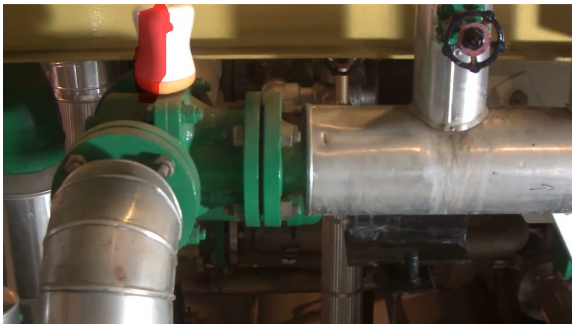
(c) Target image (black coat)

(d) Detected object (black coat)

Figure 7.3: Examples of successfully detected abandoned objects.

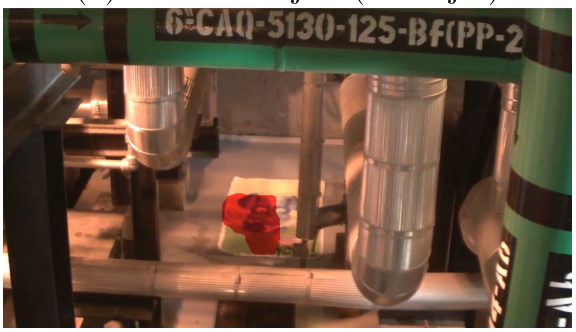


(a) Target image (white jar)



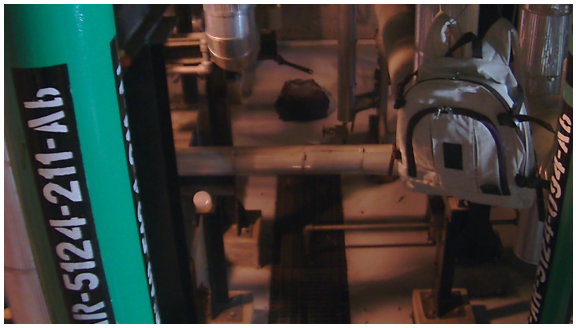
(b) Detected object (white jar)

(c) Target image (towel)

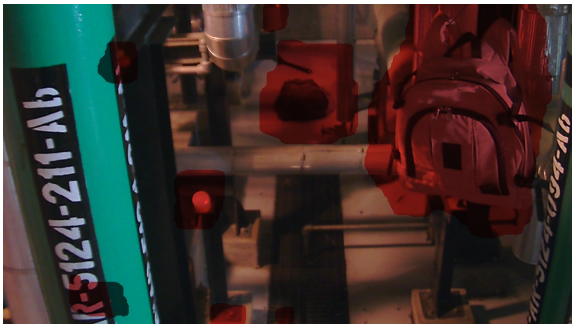


(d) Detected object (towel)

Figure 7.4: Other examples of successfully detected abandoned objects.



(a) Target image
(bottle cap, bottle, string roll,
wrench, backpack and pink backpack)



(b) Detected objects
(bottle cap, bottle, string roll,
wrench, backpack and pink backpack)

(c) Target image
(pink backpack, green box, umbrella,
lamp-bulb box and bag)

(d) Detected objects
(pink backpack, green box, umbrella,
lamp-bulb box and bag)

Figure 7.5: Examples of successfully detected abandoned objects in videos containing multiple abandoned objects.

Chapter 8

Conclusions

In this thesis, we propose a technique for real-time detection of abandoned objects in a cluttered environment. We based our proposal in a group of techniques commonly employed in similar problems, such as SURF for Point of Interest detection together with RANSAC for outlier removal and homography computation for frame registration. We also used the well-known normalized cross-correlation for detecting changes among frames.

We introduced a multiresolution approach that allowed us to detect multiple abandoned objects, of different apparent sizes, not only in the same video footage, but also in the same video frames. We employed temporal filtering to eliminate false positives, and proposed a voting step to deal with false positives as well, but also to (re)introduce pixels in detection masks in order to improve the temporal consistency of the abandoned detection process, dealing with false negatives. Morphological operations were used to remove isolated small regions in the detection masks and to connect detection spots in the cases where a single objects is detected as multiple ones. Also, we have proposed a method for performing the synchronization of the reference and target videos based solely on the acquired videos, deeming the use of external signals unnecessary

Another contribution of this research is a procedure for tuning the algorithm parameters based on a training set containing single abandoned objects. We have evaluated its effectiveness using also a single-object validation set. The database used, the VDAO database, has been produced in a real cluttered surveillance environment consisting of a room with energy generators and many pipes. It is composed of 54 videos featuring a single object as well as 3 videos containing multiple objects simultaneously shown. Other 8 videos were acquired with no objects for reference purposes. The database was manually annotated with the help of a software tool.

This real world cluttered environment database, essential to the development and testing of out algorithms, is publicly available at [54]. We hope that others will use it in their researches, comparing their results with the ones presented in this

thesis.

We have performed the overall system evaluation employing three multi-object videos, using three sets of evaluation metrics. The first set of evaluation metrics used, METE, has been proposed in [65] and is based on the percentages of superposition of the detection masks and the ground truth for each object. The second set is based on the overall object detection errors on a frame-by-frame basis. We proposed this second metric for measuring the detection error rate at the object level because the positions and the dimensions of the bounding boxes of the produced detection spots were not very similar to the ones annotated in the database, this way providing large values in our METE evaluations (in METE, the smaller the measurement, the better). As what we want to measure is if the abandoned object is detect or not, the proposed metric is more suited for our application than METE. Finally, the third set is just the decision about the presence of an abandoned object in the environment that would be made by an operator watching the videos with the detection masks. In this last case, all objects in all testing set videos could be properly detected.

In brief, the original contributions of the work have been on:

i Initial video alignment using a maximum likelihood approach that precludes the use of any external trigger signals;

ii Voting process among frames to increase detection robustness.

iii A multiresolution approach in order to cope with the detection of multiple objects of different sizes in a single video. Bigger objects are detected in smaller resolutions, and progressively smaller objects are detected in progressively bigger resolutions;

iv The development and use of an annotated database, with more than 8 hours of recorded video, produced in a real world cluttered environment (an industrial plant) in order to properly train and test our algorithms;

v The development, using a training set that is representative of a given usage scenario, of a procedure for parameter tuning based on the impact of each parameter of our method in the abandoned object detection performance. The four analysed parameters were the NCC Binarization Threshold, the Temporal Filtering Length, the Voting Vector Size, and the Voting Threshold;

vi The proposal of a metric in order to evaluate the obtained results.

The previously studied techniques, shown in Chapter 2, employ static or PTZ cameras, use external triggers for video alignment, process data offline, are not suited to be used in cluttered environments, or have other different constraints or

requirements not present in our problem. This way, the database had to be produced, and the algorithms, parameter tuning procedure and evaluation metric had to be developed in order to allow us to deal with it.

Our abandoned object detection process, then, is composed of a video synchronization step, an image registration step, and a multiresolution approach to compare the registered frames. This multiresolution approach is composed of a step do produce a binarized NCC image, a temporal filtering, a voting step, and morphological operations (Fig. 3.2 shows a block diagram representing the whole process).

With the techniques presented in this document, we could successfully detect abandoned objects with good true positive rates and low false negative rates, with the possibility of performing this task in real time. Observing the achieved results, we believe that such a system has a good potential application in the surveillance of cluttered environments such as industrial plants. In these kinds of environments, with the help of a system using the techniques described in this thesis, their economic operation could be greatly improved. Also, allowing automated operations to be performed would help to greatly minimize labor risks to which people would normally be subjected to in their works. This way, automatic detection of abandoned objects with a moving camera using multiscale video analysis could help in the protection of human lives.

Bibliography

[1] DORE, A., SOTO, M., REGAZZONI, C. S. "Bayesian Tracking for Video Analytics", *IEEE Signal Processing Magazine*, v. 27, n. 5, pp. 46–55, September 2010.

[2] SALIGRAMA, V., KONRAD, J., JODOIN, P.-M. "Video Anomaly Identification", *IEEE Signal Processing Magazine*, v. 27, n. 5, pp. 18–33, September 2010.

[3] SUBUDHI, B. N., NANDA, P. K., GHOSH, A. "A Change Information Based Fast Algorithm for Video Object Detection and Tracking", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 21, n. 7, pp. 993–1004, July 2011.

[4] TIAN, Y., FERIS, R., LIU, H., et al. "Robust Detection of Abandoned and Removed Objects in Complex Surveillance Videos", *IEEE Transactions on Systems, Man, and Cybernetics*, v. 41, n. 5, pp. 565–576, 2011.

[5] JODOIN, P.-M., SALIGRAMA, V., KONRAD, J. "Behavior Subtraction", *IEEE Transactions on Image Processing*, v. 21, n. 9, pp. 4244–4255, September 2012.

[6] CHENG, L., GONG, M., SCHUURMANS, D., et al. "Real-Time Discriminative Background Subtraction", *IEEE Transactions on Image Processing*, v. 20, n. 5, pp. 1401–1414, May 2011.

[7] TOMIOKA, Y., TAKARA, A., KITAZAWA, H. "Generation of an Optimum Patrol Course for Mobile Surveillance Camera", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 22, n. 2, pp. 216–224, February 2012.

[8] KONG, H., AUDIBERT, J.-Y., PONCE, J. "Detecting Abandoned Objects with a Moving Camera", *IEEE Transactions on Image Processing*, v. 19, n. 8, pp. 2201–2210, August 2010.

[9] CHEN, Y.-M., BAJIE, I. V. "A Joint Approach to Global Motion Estimation and Motion Segmentation from a Coarsely Sampled Motion Vector Field", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 21, n. 9, pp. 1316–1328, September 2011.

[10] SUHR, J. K., JUNG, H. G., LI, G., et al. "Background Compensation for Pan-Tilt-Zoom Cameras Using 1-D Feature Matching and Outlier Rejection", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 21, n. 3, pp. 371–377, March 2011.

[11] XUE, K., OGUNMAKIN, G., LIU, Y., et al. "PTZ Camera-Based Adaptive Panoramic and Multi-Layered Background Model". In: *18th IEEE International Conference on Image Processing*, Brussels, Belgium, September 2011.

[12] KIM, J., YE, G., KIM, D. "Moving Object Detection Under Free-Moving Camera". In: *17th IEEE International Conference on Image Processing*, Hong Kong, September 2010.

[13] XIE, Y., LIN, L., JIA, Y. "Tracking Objects with Adaptive Feature Patches for PTZ Camera Visual Surveillance". In: *20th International Conference on Pattern Recognition*, pp. 1739–1742, Istanbul, Turkey, August 2010.

[14] KUNDU, A., JAWAHAR, C. V., KRISHNA, K. M. "Realtime Moving Object Detection from a Freely Moving Monocular Camera". In: *IEEE International Conference on Robotics and Biomimetics*, pp. 1635–1640, Tianjin, China, December 2010.

[15] DESOUZA, G. N., KAK, A. C. "Vision for Mobile Robot Navigation: A Survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 2, pp. 237–267, February 2002.

[16] DA SILVA, A. F., THOMAZ, L. A., CARVALHO, G., et al. "An Annotated Video Database for Abandoned-Object Detection in a Cluttered Environment". In: *International Telecommunications Symposium*, Sao Paulo, Brazil, August 2014. DOI: 10.1109/ITS.2014.6947966.

[17] ZHOU, D., ZHANG, H. "Modified GMM Background Modeling and Optical Flow for Detection of Moving Objects". In: *IEEE International Conference on Systems, Man and Cybernetics*, pp. 2224–2229, Waikoloa, USA, October 2005.

[18] YU, Y., ZHOU, C., HUANG, L., et al. "A Moving Target Detection Algorithm Based on the Dynamic Background". In: *International Conference on Computational Intelligence and Software Engineering*, pp. 1–5, Wuhan, China, December 2009.

[19] PEIJIANG, C. "Moving Object Detection Based on Background Extraction". In: *International Symposium on Computer Network and Multimedia Technology*, pp. 1–4, Wuhan, China, January 2009.

[20] SU, S.-T., CHEN, Y.-Y. "Moving Object Segmentation Using Improved Running Gaussian Average Background Model". In: *Digital Image Computing: Techniques and Applications*, pp. 24–31, Canberra , Australia, December 2008.

[21] LIPTON, A. J., FUJIYOSHI, H., PATIL, R. S. "Moving Target Classification and Tracking from Real-time Video". In: *Proceedings of the Fourth IEEE Workshop on Applications of Computer Vision*, pp. 8–14, Princeton, USA, October 1998.

[22] NIU, L., JIANG, N. "A Moving Objects Detection Algorithm Based on Improved Background Subtraction". In: *Eighth International Conference on Intelligent Systems Design and Applications*, pp. 604–607, Kaohsiung , Taiwan, November 2008.

[23] CHENG, F.-C., HUANG, S.-C., RUAN, S.-J. "Illumination-Sensitive Background Modeling Approach for Accurate Moving Object Detection", *IEEE Transactions on Broadcasting*, v. 57, n. 4, pp. 794–801, December 2011.

[24] LI, Q.-Z., HE, D.-X., WANG, B. "Effective Moving Objects Detection Based on Clustering Background Model for Video Surveillance". In: *Congress on Image and Signal Processing*, pp. 656–660, Sanya , China, May 2008.

[25] HU, F.-Y., ZHANG, Y.-N., YAO, L. "An Effective Detection Algorithm for Moving Object with Complex Background". In: *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, pp. 5011–5015, Guangzhou , China, August 2005.

[26] HU, H., LI, Z., QU, Z., et al. "Vision-based Moving Objects Detection with Background Modeling". In: *International Conference on Measuring Technology and Mechatronics Automation*, pp. 436–439, Zhangjiajie, China, April 2009.

[27] LEI, T., FAN, Y., LI, L. "The Algorithm of Moving Human Body Detection Based on Region Background Modeling". In: *International Symposium on Computer Network and Multimedia Technology*, pp. 1–4, Wuhan , China, January 2009.

[28] FRIEDMAN, N., RUSSELL, S. "Image segmentation in Video Sequences: A Probabilistic Approach". In: *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pp. 175–181, 1997.

[29] STAUFFER, C., GRIMSON, W. "Adaptive Background Mixture Models for Real-time Tracking". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Fort Collins, USA, June 1999.

[30] ZHANG, J., CHEN, C. H. "Moving Objects Detection and Segmentation In Dynamic Video Backgrounds". In: *IEEE Conference on Technologies for Homeland Security*, pp. 64–69, Woburn, USA, May 2007.

[31] RIDDER, C., MUNKELT, O., KIRCHNER, H. "Adaptive Background Estimation and Foreground Detection using Kalman-Filtering". pp. 193–199, 1995.

[32] XIA, Y., NING, S., SHEN, H. "Moving Targets Detection Algorithm Based on Background Subtraction and Frames Subtraction". In: *2nd International Conference on Industrial Mechatronics and Automation*, pp. 122–125, Wuhan , China, May 2010.

[33] TOYAMA, K., KRUMM, J., BRUMITT, B., et al. "Wallflower: Principles and Practice of Background Maintenance". In: *International Conference on Computer Vision*, pp. 255–261, Kerkyra , Greece, September 1999.

[34] BORAGNO, S., BOGHOSSIAN, B., MAKRIS, D., et al. "Object Classification for Real-Time Video-Surveillance Applications". In: *5th International Conference on Visual Information Engineering*, pp. 192–197, Xian, China, July 2008.

[35] FISCHLER, M. A., BOLLES, R. C. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", *Communications of the ACM*, v. 24, n. 6, pp. 381–395, June 1981.

[36] MURRAY, D., BASU, A. "Motion Tracking with an Active Camera", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 16, n. 5, pp. 449–459, May 1994.

[37] FEGHALI, R., MITICHE, A. "Spatiotemporal Motion Boundary Detection and Motion Boundary Velocity Estimation for Tracking Moving Objects With a Moving Camera: A Level Sets PDEs Approach With Concurrent Camera Motion Compensation", *IEEE Transactions on Image Processing*, v. 13, n. 11, pp. 1473–1490, November 2004.

[38] HUANG, C.-M., CHEN, Y.-R., FU, L.-C. "Real-Time Object Detection and Tracking on a Moving Camera Platform". In: *International Conference on Control, Automation and Systems*, pp. 717–722, Fukuoka, Japan, August 2009.

[39] BERRABAH, S., CUBBER, G. D., ENESCU, V., et al. "MRF-Based Foreground Detection in Image Sequences from a Moving Camera". In: *IEEE International Conference on Image Processing*, pp. 1125–1128, Atlanta, USA, October 2006.

[40] KUNDU, A., KRISHNA, K. M., SIVASWAMY, J. "Moving Object Detection by Multi-View Geometric Techniques from a Single Camera Mounted Robot". In: *International Conference on Intelligent Robots and Systems*, pp. 4306–4312, St. Louis, USA, October 2009.

[41] KNOWLES, M. J., SPANN, M. "Motion Compensated Background Filtering for Real Time Tracking in Moving Background Sequences". In: *IEEE International Conference on Systems, Man and Cybernetics*, pp. 1059–1065, Waikoloa, USA, October 2005.

[42] ZHANG, Y., KISELEWICH, S. J., BAUSON, W. A., et al. "Robust Moving Object Detection at Distance in the Visible Spectrum and Beyond Using A Moving Camera". In: *Conference on Computer Vision and Pattern Recognition Workshop*, June 2006.

[43] ZHOU, D., WANG, L., CAI, X., et al. "Detection of Moving Targets with a Moving Camera". In: *IEEE International Conference on Robotics and Biomimetics*, pp. 677–681, Guilin, China, December 2009.

[44] LOWE, D. G. "Object Recognition from Local Scale-Invariant Features". In: *International Conference on Computer Vision*, Corfu, Greece, September 1999.

[45] LOWE, D. G. "Local Feature View Clustering for 3D Object Recognition". In: *Conference on Computer Vision and Pattern Recognition*, Kauai, USA, December 2001.

[46] LOWE, D. G. "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision*, 2004.

[47] DA SILVA, A. F. *Determining of Camera Trajectory for Abandoned Object Detection.* M.Sc. dissertation, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil, 2015.

[48] THOMAZ, L. A. *Abandoned Object Detection Using Operator-Space Pursuit.* M.Sc. dissertation, Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil, 2015.

[49] JARDIM, E., BIAN, X., DA SILVA, E. A. B., et al. "On the Detection of Abandoned Objects with a Moving Camera Using Robust Subspace Recovery and Sparce Representation", Accepted for presentation at IEEE International Conference on Acoustics, Speech, and Signal Processing, April 2015.

[50] BIAN, X., KRIM, H. "Optimal Operator Space Pursuit: A Framework for Video Sequence Data Analysis". In: *11th Asian Conference on Computer Vision*, pp. 760–769, Daejeon, Korea, November 2012.

[51] BIAN, X., KRIM, H. "Video-Based Human Activities Analysis: An Operator-Based Approach". In: *20th International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision*, Pilsen, Czech Republic, June 2012.

[52] BIAN, X., KRIM, H. "Robust Subspace Recovery via Bi-Sparsity Pursuit". [arXiv:1403.8067v2], April 2014. Available at `http://arxiv.org/abs/1403.8067`.

[53] "IRobot Roomba Vacuum Cleaning Robot". [Online]. `http://www.irobot.com/For-the-Home/Vacuum-Cleaning/Roomba.aspx`.

[54] "VDAO - Video Database of Abandoned Objects in a Cluttered Industrial Environment". [Online]. Available at `http://www.smt.ufrj.br/~tvdigital/database/objects`.

[55] DEITEL, P. J., DEITEL, H. M. *C++ Como Programar.* 3rd ed. Porto Alegre, Brazil, Bookman, 2001.

[56] BLANCHETTE, J., SUMMERFIELD, M. *C++GUI Programming with Qt 4.* 1st ed. Massachusetts, USA, Prentice-Hall, 2006.

[57] LAGANIÈRE, R. *OpenCV 2 Computer Vision Application Programming Cookbook.* 1st ed. Birmingham, UK, Packt Publishing, 2011.

[58] BAY, H., ESS, A., TUYTELAARS, T., et al. "Speeded-Up Robust Features (SURF)", *Computer Vision and Image Understanding*, v. 110, n. 3, pp. 346–359, 2008.

[59] KUCHARCZAK, F., DA SILVA, A. F., THOMAZ, L. A., et al. "Comparison and Optimization of Image Descriptors for Real-Time Detection of Abandoned Objects". In: *Simpósio de Processamento de Sinais da UNICAMP*, Campinas, Brazil, September 2014.

[60] CARVALHO, G., DE OLIVEIRA, J. F. L., DA SILVA, E. A. B., et al. "Um Sistema de Monitoramento para Deteccao de Objetos em Tempo Real Empregando Camera em Movimento". In: *XXXI Simposio Brasileiro de Telecomunicacoes*, Fortaleza, Brazil, September 2013.

[61] LEUTENEGGER, S., CHLI, M., SIEGWART, R. Y. "BRISK: Binary Robust Invariant Scalable Keypoints". In: *IEEE International Conference on Computer Vision*, pp. 2548–2555, Barcelona, Spain, November 2011.

[62] ALAHI, A., ORTIZ, R., VANDERGHEYNST, P. "FREAK: Fast Retina Keypoint". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 510–517, Providence, USA, June 2012.

[63] HARTLEY, R., ZISSERMAN, A. *Multiple View Geometry in Computer Vision.* 2nd ed. Cambridge, U.K, Cambridge University Press, 2003.

[64] SOILLE, P. *Morphological Image Analysis: Principles and Applications.* 2nd ed. Berlin, Springer, 2003.

[65] NAWAZ, T., POIESI, F., CAVALLARO, A. "Measures of Effective Video Tracking", *IEEE Transactions on Image Processing*, v. 23, n. 1, pp. 376–388, January 2014.