Feature-Aided Multitarget Tracking for Optical Belt Sorters

Tobias Kronauer[*], **Florian Pfaff**[*], **Benjamin Noack**[*], **Wei Tian**[†], **Georg Maier**[‡] and **Uwe D. Hanebeck**[*]

[*]Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Karlsruhe Institute of Technology (KIT), Germany
tobias.kronauer92@gmx.de, florian.pfaff@kit.edu, benjamin.noack@kit.edu, uwe.hanebeck@ieee.org

[†]Institute of Measurement and Control Systems (MRT), Karlsruhe Institute of Technology (KIT), Germany
wei.tian@kit.edu

[‡]Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB), Germany
georg.maier@iosb.fraunhofer.de

***Abstract*—Industrial optical belt sorters are highly versatile in sorting bulk material or food, especially if mechanical properties are not sufficient for an adequate sorting quality. In previous works, we could show that the sorting quality can be enhanced by replacing the line scan camera, which is normally used, with an area scan camera. By performing multitarget tracking within the field of view, the precision of the utilized separation mechanism can be enhanced. The employed kinematics-based multitarget tracking crucially depends on the ability to associate detection hypotheses of the same particle across multiple frames.**

In this work, we propose a procedure to incorporate the visual similarity of the detected particles into the kinematics-based multitarget tracking that is generic and evaluates the visual similarity independent of the kinematics. For evaluating the visual similarity, we use the Kernelized Correlation Filter, the Large Margin Nearest Neighbor method and the Normalized Cross Correlation. Although no clear superiority for any of the visual similarity measures mentioned above could be determined, an improvement of all considered error metrics was attained.

***Index Terms*—feature-aided multitarget tracking, industrial optical belt sorters, metric learning, visual tracking.**

I. Introduction

In times of global trade, the amount of bulk material that needs to be handled keeps increasing. The amount of worldwide maritime freight of iron ore, grain, coal, bauxite, and phosphate increased from 448 million tons in 1970 to 3172 millions tons in 2016 [1]. Furthermore, the annual volume of transport and handling of bulk material is estimated to be 10 billion dollars by consuming 10% of the global energy production [2].

In order to decrease economic and ecological costs, efficient sorting is of outstanding importance. For the separation, mechanical properties such as the density or the shape of the bulk material can often be used. If an adequate sorting quality cannot be achieved based on mechanical properties, optical belt sorters can be used as an alternative. Such sorters base the separation on the visual appearance of the bulk material.

Industrial optical belt sorters, schematically shown in Fig. 1, are equipped with a line scan camera that detects the applied particles. After the particles are detected, their locations and times of arrival in front of an array of compressed-air nozzles are predicted by the data processing unit. Given the prediction, respective valves are activated, which emit air blasts that separate the particles into two classes, e.g., based on color or texture. For the prediction, two assumptions are made. First, the particles' movements follow the transport direction of the belt. Second, a constant time delay between detection and activation of the nozzles is assumed. However, these two assumptions do not hold in actual applications, since the particles' movements are erratic. Therefore, the valves may be activated at an incorrect point in time and even wrong nozzles may be activated. In previous works, we replaced the line scan camera with an area scan camera and were able to show that the sorting quality can be increased by performing multitarget tracking within the field of view of the camera [4], [5].

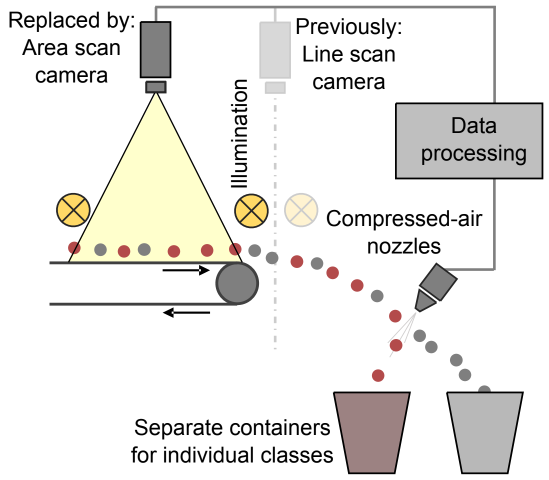

Replaced by: Area scan camera
Previously: Line scan camera
Illumination
Data processing
Compressed-air nozzles
Separate containers for individual classes

Figure 1: The new set-up with an area scan camera. The previously used line scan camera is also visualized. Figure taken from [3].

This paper is organized as follows: We start by explaining the multitarget tracking that is currently used, followed by a problem formulation. In Sec. II, we describe the visual similarity measures chosen for evaluating the visual similarity of the detected particles. The procedure to incorporate the visual

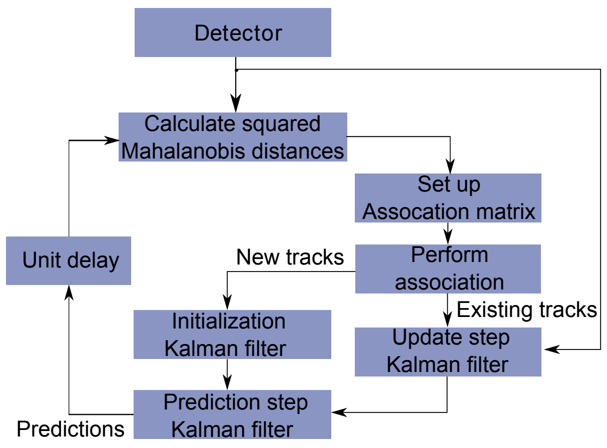

Detector
Calculate squared Mahalanobis distances
Set up Assocation matrix
Perform association
New tracks
Existing tracks
Unit delay
Initialization Kalman filter
Update step Kalman filter
Prediction step Kalman filter
Predictions

Figure 2: Schematic illustration of the kinematics-based multi-target tracking. The time indices are omitted.

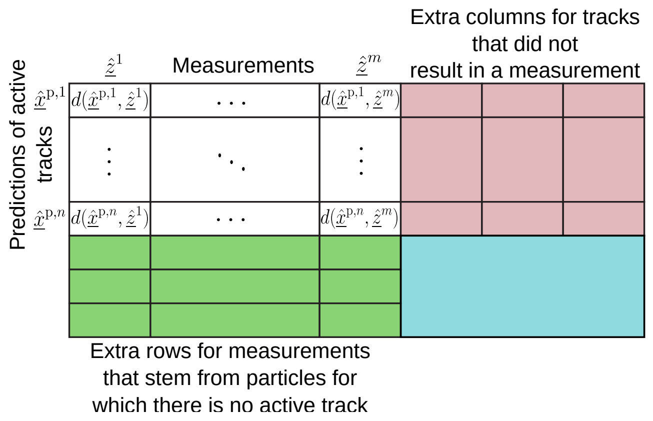

Extra columns for tracks that did not result in a measurement
$\hat{\underline{z}}^1$
Measurements
$\hat{\underline{z}}^m$
Predictions of active tracks
$\hat{\underline{x}}^{\mathrm{p},1}$
$\hat{\underline{x}}^{\mathrm{p},n}$
$d(\hat{\underline{x}}^{\mathrm{p},1}, \hat{\underline{z}}^1)$
$d(\hat{\underline{x}}^{\mathrm{p},1}, \hat{\underline{z}}^m)$
$d(\hat{\underline{x}}^{\mathrm{p},n}, \hat{\underline{z}}^1)$
$d(\hat{\underline{x}}^{\mathrm{p},n}, \hat{\underline{z}}^m)$
Extra rows for measurements that stem from particles for which there is no active track

Figure 3: Association matrix that is used to perform the data association. Figure taken from [10].

similarity into the kinematics-based multitarget tracking is presented in Sec. III. The kinematics-based multitarget tracking is compared with the feature-aided multitarget tracking [6] in Sec. IV. Finally, the paper is concluded in Sec. V.

A. Employed Multitarget Tracking Algorithm

Fig. 2 shows a simplified illustration of the employed multitarget tracking algorithm that performs hard association decisions. An external detector recognizes the applied particles and provides measurements $\hat{\underline{z}}^{\mathrm{pos},j}$ with index j as centroids of the detections at each time step. The detection procedure can be subsumed to the following four successive steps: image preprocessing, segmentation, connected components analysis, and, finally, determining the centroids of the resulting contours. Segmentation is performed by binarizing the images according to predefined HSV values.

A Kalman filter is used for motion estimation of each particle. As state variables, the x- and y-coordinate of the centroid and the velocity in x- and y-direction are used. We employ a constant velocity model. In previous works, we proposed to extend the state vector by the orientation of the particles [7].

Assuming an association to be given, the prediction step of the Kalman filter is executed. In order to perform an association for a subsequent frame, let us assume that we have n measurements that need to be assigned to n tracks. In this case, an association can be seen as a permutation τ of $\{1, \ldots, n\}$. We choose the permutation that maximizes the global association likelihood [8] under the constraint that only one measurement can be assigned to each track. This is equivalent to finding the permutation τ that minimizes the negative logarithm of the likelihoods, which can be expressed as $-\sum_{i=1}^{n} \log l(\hat{\underline{z}}^{\mathrm{pos},\tau(i)}|i)$, with l denoting the likelihood that the measurement $\hat{\underline{z}}^{\mathrm{pos},\tau(i)}$ stems from track i. If Gaussian densities in Euclidean space are assumed, the sum can be decomposed into two parts: A constant part that is independent of τ and a second part that can be regarded as the sum of squared Mahalanobis distances between the predicted positions of the tracks and the measurements multiplied by a constant [8].

All additive constants and constant factors can be disregarded as they do not influence which permutation is optimal. We arrange the squared Mahalanobis distances for all predicted positions and measurements as shown in Fig. 3. The index of the track increases vertically while the index of the measurements increases horizontally. In this form, the association problem can be regarded as a linear assignment problem (LAP) that can be solved using the Hungarian Algorithm [9]. To account for particles that enter or leave the field of view of the camera, extra rows and columns are appended to the association matrix. The values for the extra entries are location-dependent [10]. Afterward, the update step of the Kalman filters is performed. The procedure described is performed at each time step.

B. Problem Formulation

At each time step t, an external detector provides m_t centroids of the detected particles as measurements. For the sake of clarity, the time index will be omitted in the rest of the paper. Although misdetections and (temporary) occlusions are neglected within this work, the detector often misclassifies *colliding particles*. If two particles collide in one image, they are classified as a single particle. The detector is assumed to be given and is not part of this work.

Due to the high velocity of the belt of up to $1.5\,\mathrm{m/s}$, the particles' displacements between two successive time steps are large. At each time step, the association assigns a subset of the m measurements to n existing tracks. For particles, that newly enter the field of view of the camera, new tracks shall be created. Tracks shall be deleted for particles leaving the field of view of the camera, hence, not resulting in a measurement.

Although the kinematics-based multitarget tracking results in only a few false assignments and thus in a high tracking and sorting quality, false assignments can result in incorrect sorting decisions. This can happen in situations with lots of particle collisions or large variations in the particles' shapes leading to an increase in system noise. Since the activation of the correct nozzles crucially depends on correct assignments, reducing the false assignments to a minimum is essential. The idea of this paper is to incorporate visual cues of the applied particles into a feature-aided multitarget tracking approach.

II. Visual Similarity Measures

In this section, we describe the approaches used in this work to assess the visual similarity. Since the multitarget tracking problem can be regarded as a visual tracking task, we briefly explain visual tracking in Sec. II-A and point out the differences to the employed kinematics-based multitarget tracking. Afterward, we introduce the Kernelized Correlation Filter in Sec. II-B. This is followed by the Large Margin Nearest Neighbor method used in Metric Learning in Sec. II-C and the Normalized Cross Correlation in Sec. II-D.

For each visual similarity measure, we conclude the subsection by assuming that the visual similarity between the feature vector $\hat{\underline{z}}^{\text{vis}}$ of a yet unassigned particle and the feature vector $\hat{\underline{x}}^{\text{vis}}$ of an arbitrary track i shall be calculated. The particle that was assigned to track i in the previous time step is represented by $\hat{\underline{x}}^{\text{vis}}$. The feature vector crucially depends on the visual similarity and can also be multidimensional.

A. Visual Tracking

Visual tracking [11], [12] aims at tracking objects in a sequence of images using visual cues. There are two categories of approaches [12]: Generative tracking [11] and Tracking-By-Detection (TBD) that aims at distinguishing the target from the local background by using a classifier [13]. TBD trackers outperform most generative tracking approaches [11].

TBD trackers typically use an external detector that initializes the target object by defining a bounding box. Next, a classifier is trained online by sampling and labeling the surroundings of the target object. Within a specified search region, the previously trained classifier then determines the position of the target object in the next image by choosing the image patch that maximizes the classification score. The search region is usually based on a motion model. The different TBD algorithms mainly differ in the classifier being used, its training (i.e., sampling and labeling) and the definition of the search region.

In the context of TBD, the detector is used for the initialization but not for the succeeding time steps which is in contrast to our approach. Further, the definition of a search region has two drawbacks. First, if the motion model is not valid, and hence the search region is falsely defined, the particle might actually be outside the search region. This results in a false assignment. Therefore, we define the search region by all particle detections in the current time step. Second, if an association is performed based on the combination of the squared Mahalanobis distances with the classification results, an independent plausibility check with the kinematic model is performed.

B. Kernelized Correlation Filter

The KCF is a TBD approach and was initially introduced in [14] and yields superior results compared with competing tracking approaches [14]. The core of the KCF consists of the classifier training and the detection. As a classifier, ridge regression is used, which is a linear regression with a regularization term [15]. By modeling all translations of an image patch as circulant, the training samples can be represented by a circulant data matrix. The output of the classifier is used for the visual similarity. The following introduction is based on [14]. All explanations apply for onedimensional feature vectors with one channel. The notions can be generalized to multidimensional feature vectors with multiple channels, e.g., color images.

Let us assume that the target is represented by an $n \times 1$ feature vector $\underline{x}^\top$, which is called *base sample* and serves as a positive training sample for the classifier. All translations of the base sample serve as training samples (with their labels calculated based on the translations steps) and are modeled by the data matrix

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \\ x_n & x_1 & x_2 & \dots & x_{n-1} \\ x_{n-1} & x_n & x_1 & \dots & x_{n-2} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ x_2 & x_3 & x_4 & \dots & x_1 \end{bmatrix} . \tag{1}$$

The first row of $\mathbf{X}$ denotes the base sample $\underline{x}$. A translation of the base sample by u elements is represented by $\mathbf{P}^u \underline{x}$, with $\mathbf{P}^u$ being the permutation matrix, and can be found in the u-th row of $\mathbf{X}$. Since the data matrix is periodic, all possible translations (including $u \geq n$) are accounted for. As can be seen in (1), translating the base sample by u elements corresponds to wrapping the last u elements $x_{n-u+1:n}$ around. In reality, new unknown values are expected. Therefore, the use of a bigger image patch for some padding and a cosine window is proposed [14]. We follow this idea later on.

Training a linear classifier $f(\underline{z}) = \underline{w}^\top \underline{z}$ corresponds to determining the parameter vector $\underline{w}$ by minimizing the loss function $\sum_i (f(\underline{x}_i) - y_i)^2 + \lambda \left\| \underline{w} \right\|^2$ with λ denoting the regularization parameter and y_i the regression value of the training sample $\underline{x}_i$. The closed form solution $\underline{w} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \underline{y}$ can be reformulated using the circulant structure of $\mathbf{X}$ that can be expressed as $\mathbf{X} = \mathbf{F} \operatorname{diag}(\underline{x}_f) \mathbf{F}^{\text{H}}$ with $\mathbf{F}$ being the Discrete Fourier transform (DFT) matrix, $\underline{x}_f$ being the DFT of the base sample and the superscript $^{\text{H}}$ indicating the Hermitian conjugate. The index f denotes the representation in the frequency domain. Inserting this diagonalization into the closed-form solution of the loss function, we can reformulate this as

$$\underline{w} = \mathcal{F}^{-1} \left(\frac{\underline{x}_f^* \odot \underline{y}_f}{\underline{x}_f^* \odot \underline{x}_f + \lambda} \right) , \tag{2}$$

with $\mathcal{F}^{-1}$ denoting the inverse Discrete Fourier transform, the superscript $*$ indicating the complex conjugate and the operator $\odot$ as an element-wise multiplication. The key advantage of this formula is that it can be evaluated in $\mathcal{O}(n)$ as only element-wise multiplications and divisions are involved. The DFT operations can be performed in $\mathcal{O}(n \log n)$, whereas the naïve solution for the original version is in $\mathcal{O}(n^3)$.

Similar benefits can be concluded for the nonlinear regression using the kernel trick. The parameter vector $\underline{w} = \sum_{i=1}^{n} \alpha_i \varphi(\underline{x}_i)$ is represented by a linear combination of transformed feature vectors $\varphi(\underline{x}_i)$ of the training samples. Therefore, the ridge regression is evaluated such that $f(\underline{z}) = \sum_{i=1}^{n} \alpha_i \kappa(\underline{z}, \underline{x}_i)$ with $\underline{z}$ denoting the base sample of the candidate, κ being the kernel function, and $\underline{\alpha} = (K + \lambda \mathbf{I})^{-1} \underline{y}$ with kernel matrix K. For particular kernels such as Gaussian, polynomial or linear kernels, the kernel matrix is circulant as well. It can be proven, that $\kappa(\underline{z}, \underline{x}_i)$ and K can be represented by the respective base samples $\underline{z}_f$ and $\underline{x}_f$ in the frequency domain and the kernel function κ. All translations of $\underline{z}$ are stored in the vector $\underline{f}(\underline{z})$.

Aside from the computational benefits, the KCF actually learns the visual appearance of the target object. The use of the kernel allows the classification in a nonlinear feature space.

The visual similarity between feature vector $\underline{\hat{z}}^{\text{vis}}$ and $\underline{\hat{x}}^{\text{vis}}$ is calculated as follows:

1) Evaluate the classifier $\underline{f}(\underline{\hat{z}}^{\text{vis}}) = \mathcal{F}^{-1}(\underline{k}_{\hat{x}\hat{z},f} \odot \underline{\alpha}_f)$ with $\underline{\alpha}_f$ denoting the coefficient vector for the corresponding track from the previous time step. The vector $\underline{k}_{\hat{x}\hat{z},f}$ can be obtained by calculating the DFT of $\underline{\hat{z}}^{\text{vis}}$ and $\underline{\hat{x}}^{\text{vis}}$ with the kernel function κ. Both feature vectors are considered as base samples. The kernel function is defined prior to the tracking. The maximum value of the elements of $\underline{f}$ yields the visual similarity.
2) Train the classifier by calculating the coefficient vector $\underline{\alpha}^i = \mathcal{F}^{-1}\left(\frac{\underline{y}_f}{\underline{k}_{\hat{x}\hat{x},f} + \lambda}\right)$. The constant regularization parameter λ is defined prior to the tracking. The regression vector $\underline{y}$ follows a normal distribution as in [14] with a maximum of one in the center of the target. The rate of flattening is provided by the standard deviation, which is a tenth of the feature vector length.

C. Large Margin Nearest Neighbor method

The Large Margin Nearest Neighbor (LMNN) method, initially introduced in [16] is a state-of-the-art approach in the field of metric learning [17], [18]. The goal of metric learning is to learn a distance-like function in a feature space. Often, the learned metric is subsequently used to improve a certain quality measure of a classifier that is based on distance calculation such as the k nearest neighbor classifier (kNN).

The following introduction to LMNN is based on [16]. The aim is to learn a matrix $\mathbf{L}$ offline that transforms the feature space. Using this matrix, a standard kNN classification is used based on the Euclidean distance $d_{\mathbf{L}} = \sqrt{(\mathbf{L}\underline{\hat{x}}^{\text{vis}} - \mathbf{L}\underline{\hat{z}}^{\text{vis}})^{\top}(\mathbf{L}\underline{\hat{x}}^{\text{vis}} - \mathbf{L}\underline{\hat{z}}^{\text{vis}})}$ in the linear transformed feature space. The transformation is given by the matrix multiplication of the feature vectors $\underline{\hat{x}}^{\text{vis}}$ and $\underline{\hat{z}}^{\text{vis}}$ with $\mathbf{L}$. Visualized in Fig. 4, the training pursues two aims:

1) Out of the neighbors that share the same label, pull together the k closest ones (the *target neighbors*).
2) Push the *impostors* of $\underline{x}_i$ away. The impostors denote all the training samples that are within the region spanned by the target neighbors plus a margin.

The target neighbors and impostors are determined at the beginning of the training depending on the parameter k and the margin. The margin is usually set to the unit margin. Prior knowledge (e.g. a similarity graph) can be used to determine the

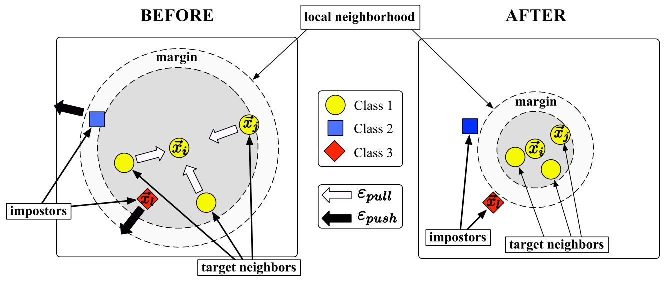


Figure 4: The goal of the training of the Large Margin Nearest Neighbor is to pull the k target neighbors together and to push the impostors away. The target neighbors and the impostors do not change within the training process. Figure taken from [16].

closest neighbors of $\underline{x}_i$. If unavailable, the target neighbors are selected based on Euclidean distances. This is the approach we choose. Although $\mathbf{L}$ is trained by considering local information, the transformation affects all class samples in the same way.

Calculating the visual similarity of the feature vectors $\underline{\hat{x}}^{\text{vis}}$ and $\underline{\hat{z}}^{\text{vis}}$ comprises the following steps:

1) Train transformation matrix $\mathbf{L}$ offline. Beside the labeled feature vectors, the learning rate required for gradient descent and the number of target neighbors k are required as parameters for the training. Each track in the training data set corresponds to one class.
2) Transform the feature vectors by calculating the matrix product $\mathbf{L}\underline{\hat{x}}^{\text{vis}}$ and $\mathbf{L}\underline{\hat{z}}^{\text{vis}}$, respectively.
3) Finally, calculate the Euclidean distance $d_{\mathbf{L}} = \sqrt{(\mathbf{L}\underline{\hat{x}}^{\text{vis}} - \mathbf{L}\underline{\hat{z}}^{\text{vis}})^{\top}(\mathbf{L}\underline{\hat{x}}^{\text{vis}} - \mathbf{L}\underline{\hat{z}}^{\text{vis}})}$.

D. Zero Mean Normalized Cross Correlation

The correlation algorithm is a common and simple approach to compare two image patches. Among others, they are used in the aforementioned generative tracking. The following explanation is based on [19].

Let us assume the two image patches I_1 and I_2 are to be compared. The goal is to find the point in I_2 that corresponds to point (u_0, v_0) in I_1. There is a displacement denoted by the vector $\underline{d} = (d_u, d_v)$ between the two image patches.

First, an image patch is normalized by the mean of the pixel values $\bar{I}_1 = \frac{1}{n^2} \sum_u \sum_v I_1(u, v)$ with $I_1(u, v)$ denoting the grayscale pixel values at pixel (u, v) of image patch I_1 with dimension $n \times n$. Second, the pixel values are divided by the standard deviation of the pixel values of the image patch $\sigma_{I_1} = \sqrt{\sum_u \sum_v (I_1(u, v) - \bar{I}_1)^2}$. Same procedures are also applied for I_2. This yields the zero mean normalized cross correlation (ZNCC, often also referred to as *Normalized Cross Correlation*) in the following form:

$$\text{ZNCC}(I_1, I_2) = \frac{\sum_u \sum_v (I_1(u, v) - \bar{I}_1) \cdot (I_2(u, v) - \bar{I}_2)}{\sigma_{I_1} \sigma_{I_2}} \tag{3}$$

The visual similarity of $\underline{\hat{x}}^{\text{vis}}$ and $\underline{\hat{z}}^{\text{vis}}$ is calculated in the following three steps:

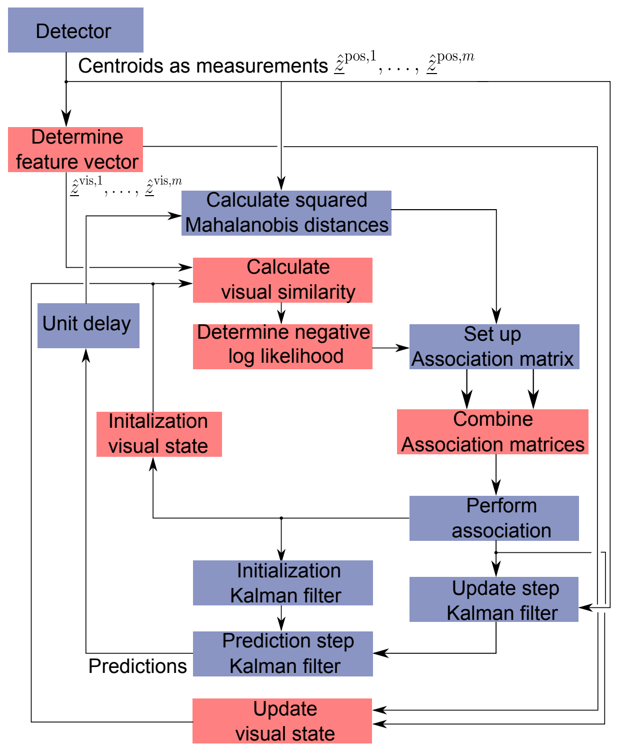

Detector
Centroids as measurements $\underline{\hat{z}}^{\text{pos},1}, \dots, \underline{\hat{z}}^{\text{pos},m}$
Determine feature vector
$\underline{\hat{z}}^{\text{vis},1}, \dots, \underline{\hat{z}}^{\text{vis},m}$
Calculate squared Mahalanobis distances
Calculate visual similarity
Unit delay
Determine negative log likelihood
Set up Association matrix
Initalization visual state
Combine Association matrices
Perform association
Initialization Kalman filter
Update step Kalman filter
Prediction step Kalman filter
Predictions
Update visual state

Figure 5: Illustration of the proposed feature-aided multitarget tracking. The red boxes indicate the required steps for incorporating the visual similarity.

1) Calculate the Normalized Cross Correlation according to (3) with $I_1 = \underline{\hat{z}}^{\text{vis}}$ and $I_2 = \underline{\hat{x}}^{\text{vis}}$. If necessary, the images need to be converted to grayscale.
2) Save the results for all relative shifts of the two image patches in a matrix $\mathbf{Z}$.
3) The maximum value of $\mathbf{Z}$ yields the visual similarity.

III. Feature-Aided Multitarget Tracking

In this section, we present an approach to feature-aided multitarget tracking that uses visual information. In Sec. III-A, we develop an approach to combine the visual similarities given in form of the entries of the vision-based association matrix with the kinematics-based association matrix, which is described in Sec. I. In Sec. III-B, we outline preliminary steps before the feature-aided multitarget tracking can be evaluated.

Fig. 5 depicts the feature-aided multitarget tracking as an extension of the kinematics-based multitarget tracking in Fig. 2. For simplicity's sake, the time indices are omitted. The red boxes indicate the new steps.

As the detector only provides centroids $\underline{\hat{z}}^{\text{pos},1}, \dots, \underline{\hat{z}}^{\text{pos},m}$ as measurements, the respective feature vectors $\underline{\hat{z}}^{\text{vis},1}, \dots, \underline{\hat{z}}^{\text{vis},m}$ need to be determined at each time step. The feature vector is extracted from an image patch defined by a bounding box whose center is defined by the obtained measurement. The size of the bounding box and the feature vector depend on the visual similarity measure.

The steps 'Calculate visual similarity' and 'Determine negative log likelihood' serve to calculate the entries of the association matrix. First, the visual similarity of all possible combinations of the feature vectors of the detected particles with the saved visual state of each track is calculated. The visual state of track i is the representation of the visual cues of the particle of track i that was assigned in the previous time step. The visual state is used to calculate the visual similarity. For design reasons, a small similarity value shall imply a larger visual similarity. To enforce this property, the negative logarithm is taken. Afterward, the visual similarity is mapped to a likelihood in step two 'Determine negative log likelihood' and the negative logarithm of the obtained likelihood is calculated.

Table I: Saved visual states and calculation formulae for the visual similarities. In order to turn the similarity values of the KCF and the ZNCC into a distance measure, the negative logarithm of the similarity values is calculated.

Visual state	**Visual similarity**	**Update visual state**
$\underline{\alpha}^i_f$	$\underline{f}(\underline{\hat{z}}^j_{\text{K}}) = \mathcal{F}^{-1}(\underline{k}_{\underline{\hat{x}}^i_{\text{K},f}\underline{\hat{z}}^j_{\text{K},f}} \odot \underline{\alpha}^i_f)$	$\underline{\alpha}^i_f = \frac{\underline{y}_f}{\underline{k}_{\underline{\hat{x}}^i_{\text{K},f}\underline{\hat{x}}^i_{\text{K},f}} + \lambda}$
$\underline{\hat{x}}^i_{\text{K},f}$	$-\log\max(\underline{f})$	$\underline{\hat{x}}^i_{\text{K},f} = \mathcal{F}(\underline{\hat{z}}^j_{\text{K}})$
$\underline{\hat{x}}^i_{\text{L}}$	$\lVert\mathbf{L}\underline{\hat{x}}^i_{\text{L}} - \mathbf{L}\underline{\hat{z}}^j_{\text{L}}\rVert^2$	$\underline{\hat{z}}^j_{\text{L}}$
$\underline{\hat{x}}^i_{\text{Z}}$	$\text{ZNCC}(\underline{\hat{x}}^i_{\text{Z}}, \underline{\hat{z}}^j_{\text{Z}})$ $-\log\max(\text{ZNCC})$	$\underline{\hat{z}}^j_{\text{Z}}$

Tab. I lists the visual state of a track and the formulae for the respective visual similarity measure. $\underline{\hat{x}}^i$ denotes the feature vector of the particle that was assigned to the track with index i in the previous time step. For better readability, we omit the superscript *vis* for the feature vectors and abbreviate each visual similarity measure with the respective first letter.

Based on the results, the vision-based association matrix is set up, which is afterward combined with the kinematics-based association matrix. These two steps are presented in Sec. III-A. If the association is solely based on the vision-based association matrix, we call this *vision-based* multitarget tracking. After the association is performed, the visual states are updated given the feature vector of the assigned particles. In the case of the LMNN and the ZNCC, the feature vectors are used. The update of the visual state of the KCF is performed by classifier training. If a new track is initialized, the visual state is initialized as well using the formulae of the update.

The size of the bounding box and the feature vector are parameters that need to be known beforehand. Steps that need to be performed prior to the tracking are described in Sec. III-B.

A. Combining the Association Matrices

In order to combine the association matrices, we proceed as in Sec. I and follow the global association likelihood approach by reformulating the likelihood $l(\underline{\hat{z}}^{\text{pos},j}|i)$ based on $l(\underline{\hat{z}}^{\text{pos},j}, \underline{\hat{z}}^{\text{vis},j}|i)$. Assuming conditional independence, the joint likelihood can be simplified to $l(\underline{\hat{z}}^{\text{pos},j}|i)l(\underline{\hat{z}}^{\text{vis},j}|i)$ [20].

Again, we assume that n measurements $\underline{\hat{z}}^{\text{pos},1}, \dots, \underline{\hat{z}}^{\text{pos},n}$ with the feature vectors $\underline{\hat{z}}^{\text{vis},1}, \dots, \underline{\hat{z}}^{\text{vis},n}$ are to be assigned to

n existing tracks. The association can be seen as a permutation τ of $\{1, \ldots, n\}$ and is performed by minimizing

$$-\sum_{i=1}^{n} \log\left(l(\hat{\underline{z}}^{\text{vis},\tau(i)}|i)\, l(\hat{\underline{z}}^{\text{pos},\tau(i)}|i) \right) \tag{4}$$

$$= -\sum_{i=1}^{n} \log l(\hat{\underline{z}}^{\text{vis},\tau(i)}|i) - \sum_{i=1}^{n} \log l(\hat{\underline{z}}^{\text{pos},\tau(i)}|i) \tag{5}$$

$$= \sum_{i=1}^{n} - \log l(\hat{\underline{z}}^{\text{vis},\tau(i)}|i) + \frac{1}{2}\sum_{i=1}^{n} \Lambda(\tau(i)) + \sum_{i=1}^{n} \varepsilon_i, \tag{6}$$

with $\Lambda(\tau(i))$ being the entries of the kinematics-based association matrix denoting the squared Mahalanobis distances between measurement $\tau(i)$ and the track i. The factor $\frac{1}{2}$ and ε_i occur due to the Gaussian densities in Euclidean space [10]. The association matrices can be combined as follows:

1) Set up vision-based association matrix by calculating $-\log l(\hat{\underline{z}}^{\text{vis},j}|i)$ for all measurement-track combinations.
2) Multiply the kinematics-based association matrix by $\frac{1}{2}$.
3) Add the association matrices.

Since the association is performed by minimizing the sum, ε_i can be omitted. In the next section, we will show how the likelihood $l(\hat{\underline{z}}^{\text{vis},j}|i)$ can be determined given the visual similarity between the feature vector $\hat{\underline{z}}^{\text{vis},j}$ and the visual state of track i. Furthermore, we will show the procedure to determine the extra rows and columns for the vision-based association matrix that allows new tracks to be initialized and disappearing tracks to be deleted. We will also show how to choose appropriate feature vectors and other parameters.

B. Required Preceding Steps

Basically, there are three steps that need to be performed before the feature-aided multitarget tracking can be evaluated. First, an appropriate parameter configuration needs to be determined. The parameters such as visual features, the kind of image preprocessing, similarity-specific parameters or the size of the bounding box depend on the visual similarity measure.

Determining the parameter configuration is done by choosing a data set for which the kinematics-based multitarget tracking results in almost no false assignments. The results thereof can thus be regarded as ground truth. Since no entries for the extra rows and columns are yet determined, we only consider time steps in which neither particles newly enter the field of view of the camera nor leave it. The association is solely performed using the vision-based association matrix. We choose the parameter configuration that yields the highest number of equal associations as the kinematics-based multitarget tracking.

Second, the likelihood needs to be determined. The aforementioned data set is used and tracks are deleted that contain false assignments or are erroneously reinitialized. These errors were manually detected. For each track, we calculate the visual similarity between the assigned particles and sort them in ascending order. The similarities being on the x-axis, we generate the empirical distribution function. The function is afterward approximated by a parametric cumulative distribution function. The derivation thereof yields the likelihood. As a third step, the entries for the extra rows and columns need to be estimated that are usually a ‘tuning parameter’ [21]. In our evaluation, we use empirically determined values.

Table II: Examined parameter configurations for the KCF.

Features	**Kernel**	**Scale factor bounding box**	**Preprocessing**
FHOG	Linear	1.5	Cosine window
Pixel values	Gaussian	2.5	Image representation
VGG16	Polynomial	3.5	

Table III: Examined parameter configurations for the LMNN.

Features	**Training data set**	**Training parameters**
HOG	33 tracks	Number of target neighbors k
Grayscale values	66 tracks	Learning rate

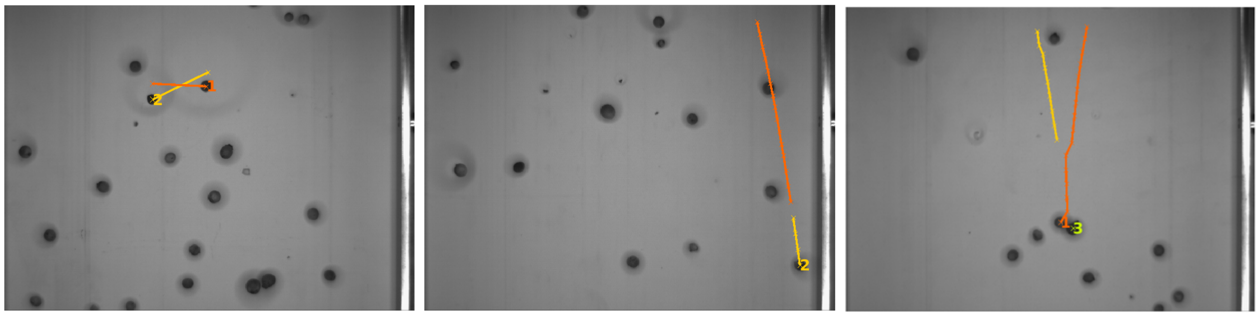
IV. Evaluation

In this section, all experiments are conducted based on a typical peppercorn sorting task. We could ascertain that if a large number (approximately 15 particles per image) of peppercorns is applied, the kinematics-based multitarget tracking results in a significant number of false assignments since the particles’ movements are erratic and hence difficult to predict. Therefore, we apply a few peppercorns to conduct the procedures described in Sec. III-B. The determined parameter configurations are presented in Sec. IV-A. For the evaluation in Sec. IV-C however, we use a large number of peppercorns. Relevant error metrics are introduced in Sec. IV-B.

A. Chosen Parameter Configurations

For all visual similarity measures, the size of the bounding box is varied. As an initial size, we use 96×96 pixels. Concerning the KCF parameters in Tab. II, we compare FHOG-features [22], pixel values, and the VGG16 [23] net. We choose the output of the 31st layer of the VGG16 as mid-level features. Different kernels with parameters as in the original implementation of [14] are compared. Furthermore, different scale factors of the bounding box are examined. Preprocessing-wise, we consider the use of a cosine window and compare grayscale value representation with color representation. Grayscale pixel values with a Gaussian kernel and a scale factor of 1.5 without a cosine window yield the best results.

As shown in Tab. III, we examine HOG features and grayscale values for training the matrix $\mathbf{L}$ using the LMNN. Since calculating $\mathbf{L}$ scales cubically with the dimension [16], we apply a PCA for dimensionality reduction on the grayscale values that keeps 99% of the total variance. We compare two training data sets containing 33 and 66 tracks, respectively. Different values for the number of target neighbors k are examined as well as the learning rate. Using 33 tracks in the training data set, $k = 4$ target neighbors, and a learning rate of 1×10^{-7} yields the highest number of equal associations. Since the Normalized Cross Correlation is applied to grayscale images, we only compare different scale factors for the



(a) Identity Switch. (b) Reinitialisation. (c) Reidentification.

Figure 6: Introduced error metrics for false assignments.

bounding box: 1, 1.5, and 2.5. The scale factor of 1.5 performs the best.

B. *Error Metrics*

In Fig. 6, we illustrate the different kinds of errors used for the evaluation. The tracks are emphasized by IDs and colors. As an *Identity Switch* depicted in Fig. 6a, we use the definition used in the MOTChallenge [24]. Let us assume we have a detected particle i at time step t, which is assigned to track j. At time step $t+1$, the next detection of the detected particle i is assigned to the existing track k with $j \neq k$. It is considered as an Identity Switch (IDSW).

In Fig. 6b, the orange track of a particular particle is falsely reinitialized with a new track-ID, depicted by the yellow track. This would indicate that the particle disappeared and a new particle entered the field of view of the camera in the middle of the image. This is accounted for by the metrics *TrackReinit*.

Lastly, *Reidentification* in Fig. 6c accounts for successful reidentifications. Since colliding particles are often falsely detected as one particle, we check whether the track-ID of a particle is the same after the collision as prior to the collision.

C. *Results*

As ground truth data set, we use peppercorns with 305 particles that result in 5375 detections across 451 time steps. The data set was manually labeled by first applying the merely kinematics-based algorithm and then correcting all errors in post-processing. For each visual similarity measure, we chose the parameter configuration that was determined in Sec. IV-A.

For the KCF, we use a logistic distribution function $p(x) = \frac{\exp\left(-\frac{x-\mu}{\beta}\right)}{\beta\left(1+\exp\left(-\frac{x-\mu}{\beta}\right)\right)}$ to approximate the likelihood with $\beta = 0.12$ and $\mu = 0.16$ and x denoting the result of the negative logarithm of the classifier function. The best fit for the likelihood of the visual similarity obtained using the LMNN is described by an inverse Gaussian distribution [25] $p(x) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\left(-\frac{\lambda}{2\mu^2 x}(x-\mu^2)\right)$ with $\mu = 4.26$ and $\lambda = 13.8$. The ZNCC is also represented by an inverse Gaussian density distribution with $\mu = 0.015$ and $\lambda = 0.016$.

In Tab. IV, we compare the feature-aided multitarget tracking with the vision-based multitarget tracking and the kinematics-based multitarget tracking using the metrics introduced in the previous section. The parameter configuration for kinematics-based multitarget tracking, i.e. employed uncertainties and motion models, is kept throughout all scenarios. The entries for the extra rows and columns of the vision-based association matrix do not change, either. However, they depend on the visual similarity measure.

Regardless of the visual similarity measure, the feature-aided multitarget tracking improves all error metrics, as can be seen in Tab. IV. Using the normalized cross-correlation as a visual similarity measure yields the highest number of reidentifications (39/49). This value is similar to the results obtained using the LMNN (36/49) or the KCF (38/49) as visual similarity measures. The kinematics-based multitarget tracking is only capable of successfully reidentifying 31 particles. In all cases, the feature-aided multitarget tracking reduces the number of track reinitializations to zero.

Therefore, incorporating the visual similarity enables a more stable tracking by reducing the number of track reinitializations and alleviates the problem of colliding particles. Although slight differences in the metrics can be observed, no clear superiority for any of the chosen similarity measures can be determined. A larger evaluation data set is hence required.

If we perform the vision-based multitarget tracking, we observe remarkable more errors in all metrics. Aside from varying approximation errors for determining the likelihood, the entries for extra rows and columns are only optimal for the feature-aided multitarget tracking. These values are influenced by the entries of the extra rows and columns of the kinematics-based multitarget tracking, which are probably suboptimal. Besides, the presumably suboptimal assumption of conditional independence of the measurements $\underline{\hat{z}}^{\text{vis},j}$ and $\underline{\hat{z}}^{\text{pos},j}$ is not valid anymore. Last but not least, motion cues appear to be far more discriminative for the association than visual cues.

It is to be said that by incorporating the visual similarity, additional information is exploited to perform the association. This inevitably leads to a less computationally efficient algorithm. The amount of additional information varies with the visual similarity measure used. In all cases, the feature vector needs to be calculated. While the Metric Learning approach applies matrix multiplications, the KCF as well as the ZNCC need to perform Fourier transforms.

Table IV: Comparing different multitarget tracking approaches. Arrows indicate whether higher or lower values are desired.

Multitarget tracking	**IDSW ↓**	**TrackReinit ↓**	**Reidentification ↑**
kinematics-based	0	9	31 / 49
KCF & Kinematics-based	0	0	38 / 49
LMNN & Kinematics-based	0	0	36 / 49
ZNCC & Kinematics-based	0	0	39 / 49
KCF	1557	169	23 / 49
LMNN	92	47	9 / 49
ZNCC	64	30	19 / 49

V. Conclusion and Outlook

In this work, we extended the kinematics-based multitarget tracking for optical belt sorters by incorporating the visual similarity of the detected particles using the Large Margin Nearest Neighbor method, the Kernelized Correlation Filter, and the Normalized Cross Correlation.

The procedure to incorporate the visual similarity into the kinematics-based multitarget tracking is universally applicable and is not limited to the visual similarity measures described in this paper. Therefore, further visual similarity measures could be investigated such as Random Online Forests or Online Boosting. Additionally, detected particles that are not assigned to the current track could explicitly be used as negative samples.

Although no clear superiority among the similarity measures could be determined, we could observe that all error metrics can be improved in all cases. By reducing the number of track reinitializations to zero, a more stable tracking is enabled. The increase in successful reidentifications shows that the problem of colliding particles can be alleviated. As we can handle more collisions, less calming of the particles may be required before the particles enter the observable area. Thus, the use of shorter belt lengths may be viable.

Since the KCF and the ZNCC also provide spatial information about the location of the maximum value, this information could be used to separate the colliding particles within the tracking process. Consequently, optimizing the detector for better particle separation is not required. In this case, the classifier training of the KCF to separate the target from the background could be beneficial.

In future works, new algorithmic approaches such as Siamese Networks [26] or Network Flow Graphs [27] could be compared with the method presented in this work.

References

[1] UNCTAD-Secretary, "Review of Maritime Transport", *United Nations Conference on Trade and Development*, 2017.

[2] J. Duran, *Sands, Powders, and Grains: An Introduction to the Physics of Granular Materials*. New York: Springer, 2000.

[3] F. Pfaff, G. Kurz, C. Pieper, G. Maier, B. Noack, H. Kruggel-Emden, R. Gruna, U. D. Hanebeck, S. Wirtz, V. Scherer, T. Längle, and J. Beyerer, "Improving Multitarget Tracking Using Orientation Estimates for Sorting Bulk Materials", in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Nov. 2017, pp. 553–558.

[4] F. Pfaff, M. Baum, B. Noack, U. D. Hanebeck, R. Gruna, T. Längle, and J. Beyerer, "TrackSort: Predictive Tracking for Sorting Uncooperative Bulk Materials", in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Sep. 2015, pp. 7–12.

[5] F. Pfaff, C. Pieper, G. Maier, B. Noack, H. Kruggel-Emden, R. Gruna, U. D. Hanebeck, S. Wirtz, V. Scherer, T. Längle, and J. Beyerer, "Simulation-based Evaluation of Predictive Tracking for Sorting Bulk Materials", in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Sep. 2016, pp. 511–516.

[6] O. E. Drummond, "On Categorical Feature-Aided Target Tracking", in *Signal and Data Processing of Small Targets 2003*, International Society for Optics and Photonics, vol. 5204, 2004, pp. 544–559.

[7] F. Pfaff, K. Li, and U. D. Hanebeck, "Association Likelihoods for Directional Estimation (to appear)", in *Proceedings of the 2019 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Taipei, Republic of China, May 2019.

[8] R. P. S. Mahler, *Statistical Multisource-Multitarget Information Fusion*. Boston, Mass.: Artech House, 2007.

[9] H. W. Kuhn, "The Hungarian Method for the Assignment Problem", *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[10] F. Pfaff, "Multitarget Tracking Using Orientation Estimation for Optical Belt Sorting", Dissertation, Karlsruhe Institute of Technology, Karlsruhe, Nov. 2018 (defended).

[11] N. Wang, J. Shi, D. Yeung, and J. Jia, "Understanding and Diagnosing Visual Tracking Systems", in *IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, pp. 3101–3109.

[12] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T.-K. Kim, "Multiple Object Tracking: A Literature Review", *arXiv:1409.7618 [cs]*, Sep. 2014.

[13] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured Output Tracking with Kernels", in *Proceedings of the 2011 International Conference on Computer Vision*, 2011, pp. 263–270.

[14] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[15] R. Rifkin, G. Yeo, T. Poggio, *et al.*, "Regularized Least-Squares Classification", *Nato Science Series Sub Series III Computer and Systems Sciences*, vol. 190, pp. 131–154, 2003.

[16] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance Metric Learning for Large Margin Nearest Neighbor Classification", in *Advances in neural information processing systems*, 2006, pp. 1473–1480.

[17] M. Guillaumin, J. Verbeek, and C. Schmid, "Is That You? Metric Learning Approaches for Face Identification", in *ICCV 2009-International Conference on Computer Vision*, IEEE, 2009, pp. 498–505.

[18] A. Bellet, A. Habrard, and M. Sebban, "A Survey on Metric Learning for Feature Vectors and Structured Data", *arXiv preprint arXiv:1306.6709*, 2013.

[19] P. Azad, T. Gockel, and R. Dillmann, *Computer Vision: Principles and Practice*, ger, 3rd ed. Aachen: Gazelle Distribution, 2011.

[20] P. Emami, P. M. Pardalos, L. Elefteriadou, and S. Ranka, "Machine Learning Methods for Solving Assignment Problems in Multi-Target Tracking", *arXiv preprint arXiv:1802.06897*, 2018.

[21] M. E. Liggins, D. L. Hall, and J. Llinas, Eds., *Handbook of Multisensor Data Fusion: Theory and Practice*, eng, 2nd ed., 22.: CRC Press, 2009.

[22] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[23] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", *arXiv:1409.1556 [cs]*, Sep. 2014.

[24] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking", *arXiv:1504.01942 [cs]*, Apr. 2015.

[25] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*, en, 2. ed, ser. Wiley Series in Probability and Statistics. Hoboken, NJ: Wiley-Interscience, 2004.

[26] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, "Learning by Tracking: Siamese CNN for Robust Target Association", in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2016, pp. 418–425.

[27] L. Zhang, Y. Li, and R. Nevatia, "Global Data Association for Multi-Object Tracking Using Network Flows", in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2008, pp. 1–8.