J Real-Time Image Proc (2019) 16:2261–2272
DOI 10.1007/s11554-017-0735-y

ORIGINAL RESEARCH PAPER

Real-time multitarget tracking for sensor-based sorting

A new implementation of the auction algorithm for graphics processing units

Georg Maier[1] · Florian Pfaff[2] · Matthias Wagner[1] · Christoph Pieper[3] · Robin Gruna[1] · Benjamin Noack[2] · Harald Kruggel-Emden[3,4] · Thomas Längle[1] · Uwe D. Hanebeck[2] · Siegmar Wirtz[3] · Viktor Scherer[3] · Jürgen Beyerer[1]

Received: 16 March 2017 / Accepted: 2 October 2017 / Published online: 20 November 2017

Abstract Utilizing parallel algorithms is an established way of increasing performance in systems that are bound to real-time restrictions. Sensor-based sorting is a machine vision application for which firm real-time requirements need to be respected in order to reliably remove potentially harmful entities from a material feed. Recently, employing a predictive tracking approach using multitarget tracking in order to decrease the error in the physical separation in optical sorting has been proposed. For implementations that use hard associations between measurements and tracks, a linear assignment problem has to be solved for each frame recorded by a camera. The auction algorithm can be utilized for this purpose, which also has the advantage of being well suited for parallel architectures. In this paper, an improved implementation of this algorithm for a graphics processing unit (GPU) is presented. The resulting algorithm is implemented in both an OpenCL and a CUDA based environment. By using an optimized data structure, the presented algorithm outperforms recently proposed implementations in terms of speed while retaining the quality of output of the algorithm. Furthermore, memory requirements are significantly decreased, which is important for embedded systems. Experimental results are provided for two different GPUs and six datasets. It is shown that the proposed approach is of particular interest for applications dealing with comparatively large problem sizes.

Keywords Linear assignment problem · Sensor-based sorting · Parallel algorithm · Graphics processing unit

1 Introduction

Sensor-based sorting is an important real-time application in the field of machine vision. Corresponding systems provide solutions to physically separate a material feed into predefined classes. In many cases, the goal is to remove low-quality parts from a material stream. Hence, the task can be understood as executing an *accept-or-reject* decision. An impression of a sensor-based sorting system is provided in Fig. 1. Typical setups include opto-pneumatic separators [1], which consist of optical sensors for perceiving the material and compressed air nozzles for physically separating it. Ordinarily, systems utilize scanning sensors, such as line scan cameras, which allow two-dimensional images of objects passing through the field of vision to be generated. For the purpose of transportation, conveyor belts, slides, or chutes are commonly used.

A general challenge in sensor-based sorting lies in minimizing the delay between perception and separation of the material. This delay mainly exists due to the required processing time of the evaluation system employed. In the case of optical sorting, the evaluation is implemented in terms of several image processing tasks. For instance, systems handle the task of preprocessing the input data,

✉ Georg Maier
georg.maier@iosb.fraunhofer.de

[1] Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB, Fraunhoferstr. 1, 76131 Karlsruhe, Germany

[2] Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

[3] Department of Energy Plant Technology (LEAT), Ruhr-Universität Bochum (RUB), Bochum, Germany

[4] Mechanical Process Engineering and Solids Processing, TU Berlin, Berlin, Germany

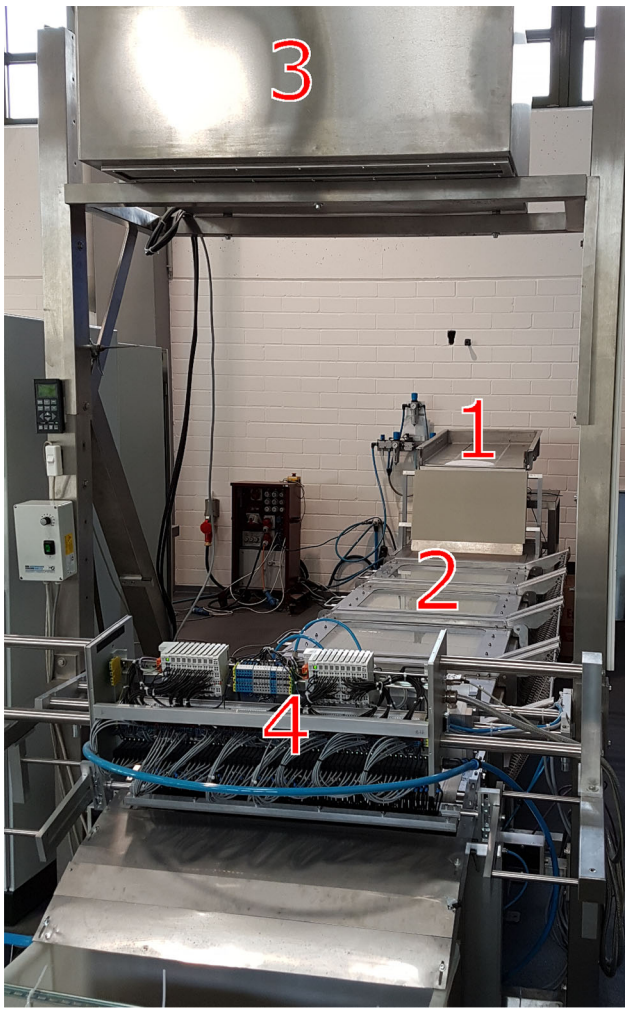


Fig. 1 Setup of an optical belt sorter. The components as chronologically passed by the material stream are shown as follows: (1) vibrating feeder (2) conveyor belt (3) sensor (4) array of compressed air nozzles

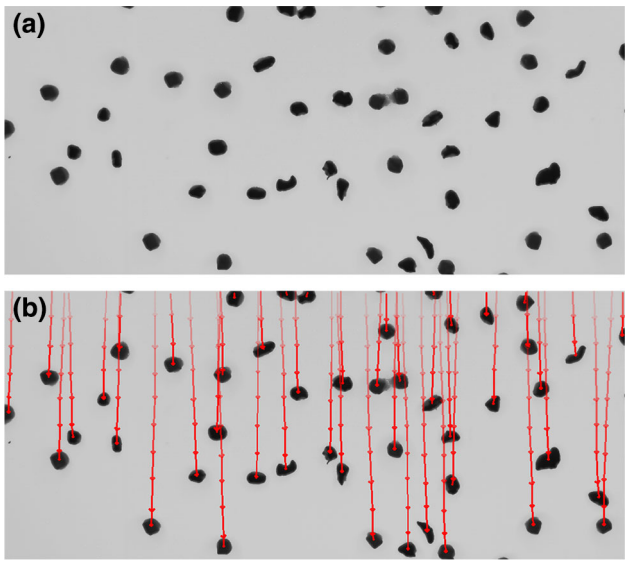


Fig. 2 Excerpt of an image recorded using a sorting system showing rubber granulate. The image was recorded using a camera of the type *Bonito CL-400* running at approximately 192 Hz and a LED ring light source. The resolution is approximately 0.13 mm per pixel. The material is captured while being transported on a conveyor belt running at 1.1 m s^{-1}. **a** Plain image **b** Image including visualization of the tracks. Each arrowed line represents the movement between two successive frames. Information from prior frames is illustrated with increasing transparency. The predictive tracking approach allows the prediction of the position of objects for subsequent frames

extracting regions of the input image that contain objects, calculating features for the individual objects, and classifying them accordingly. Minimizing this delay is a necessity for good sorting quality. An increased delay results in increased imprecision in predicting the location of an object when reaching the separation stage.

While conventional systems utilize scanning sensors, the application of area scan cameras has recently been proposed [2, 3]. Given a sufficiently high frame rate, multiple measurements of individual objects can be obtained. Utilizing multitarget tracking, the position and point in time of an individual object reaching the separation stage can be estimated more accurately [4]. An illustration of such resulting tracks is provided in Fig. 2. However, compared with conventional systems, this implies an additional burden for the evaluation systems. Furthermore, firm real-time requirements apply, since a sorting decision is useless if it is derived after the object has already passed the separation stage. Among other tasks, many established multitarget tracking systems require solving the task of assigning each obtained measurement in each new frame to a track. In order to meet the firm real-time requirements, efficient implementations are a necessity.

Bipartite graph matching, as performed in order to find the assignments between measurements and tracks, is a well-studied problem. In the field of computer vision, numerous applications exist in which such assignment tasks need to be solved. It is also used in scheduling tasks. Bipartite graph matching can be regarded as a linear assignment problem, and various algorithms to solve it exist [5]. When dealing with applications that are required to have real-time capabilities, the problem becomes even harder since corresponding solvers typically pose a high computational burden on the system whenever many correspondences are to be found.

In this paper, a real-time multitarget tracking algorithm for the computer vision task of sensor-based sorting is presented. For this purpose, an enhanced solver for the linear assignment problem is considered. More precisely, a fast realization of the auction algorithm on a graphics processing unit (GPU) is proposed. The algorithm is implemented both using the OpenCL framework, which

allows running the code on numerous state-of-the-art GPUs, and using CUDA. This allows utilization of two GPUs for experiments, further revealing the performance of the implementations. The CUDA-based implementation is based on the code published with [6]. The enhanced algorithm includes two specific improvements which significantly increase processing speed and also decrease memory usage, which is of particular interest for embedded systems. In order to demonstrate the success of this method, its applicability in multitarget tracking used for an evaluation system as included in sensor-based sorting is shown. It is also compared with other recent work in the field.

This paper is organized as follows. Following this introduction, Sect. 2 briefly reviews the linear assignment problem and the auction algorithm. Related work in the field of sensor-based sorting and fast implementations of the auction algorithm is then reviewed in Sect. 3. A general description of the multitarget tracking system considered in this paper is provided in Sect. 4. In Sect. 5, the improvements proposed in this paper are presented. These are further subject to experimental evaluation, which is provided in Sect. 6. Lastly, a conclusion is provided in Sect. 7.

2 Problem formulation

The linear assignment problem is typically formalized as follows: Let N be the number of workers and M the number of tasks. For cases when $|N| \neq |M|$, the smaller set is typically expanded such that $|N| = |M|$ holds true. Furthermore, let $x_{i,j} = 1$ in case worker i is assigned to task j and $x_{i,j} = 0$ otherwise. The cost of assigning worker i to task j is given by $a_{i,j}$. Then, the optimization problem is formulated as

$$\begin{aligned} &\min \sum_{i=1}^{N}\sum_{j=1}^{M} a_{i,j}x_{i,j} \\ &\text{s.t.} \sum_{\mathrm{j}=1}^{\mathrm{M}} \mathrm{x}_{\mathrm{i,j}} = 1 \quad \forall i = 1,\ldots,N, \\ &\sum_{i=1}^{N} x_{i,j} = 1 \quad \forall j = 1,\ldots,M. \end{aligned} \tag{1}$$

It is important to note that, due to the constraints, a one-to-one assignment is guaranteed.

The auction algorithm [7] is a popular method of solving this kind of problem. As its name already indicates, the assignments are determined by simulating an auction. Although the algorithm is used to solve a maximization problem, a minimization problem can easily be reformulated as a maximization problem, for instance by negating the cost function. The basic workflow is depicted in Fig. 3.

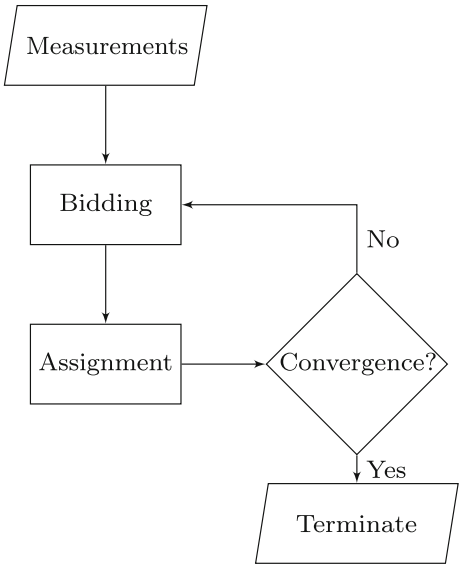


Fig. 3 Program flow of the auction algorithm

Here, N persons are competing over M objects by bidding for them. The algorithm further requires that $M \geq N$ holds true. Whenever this is not the case, the sets are swapped.

Let p_j denote the price of object j. During the bidding phase, each person finds the object of maximum value as given by $j_i \in \text{argmax}_j \{a_{i,j} - p_j\}$, where $a_{i,j}$ is the utility of object j to person i. It is important to note that $a_{i,j}$ now denotes a utility instead of a cost, which represents the reformulation from a minimization problem to a maximization problem. Having identified j_i, the person offers a bid denoted as γ_i in Eq. (2). Furthermore, v_i denotes the value of the most preferred object and w_i of the second most preferred object:

$$\begin{aligned} \gamma_i &= v_i - w_i + \epsilon \quad \text{with} \\ v_i &= \max_j \{a_{i,j} - p_j\}, \\ w_i &= \max_{j \neq j_i} \{a_{i,j} - p_j\}, \\ \epsilon &> 0. \end{aligned} \tag{2}$$

From Eq. (2) it is clear that $\gamma_i > 0$ holds. Without ϵ, there could be cases when $\gamma_i = 0$ and the algorithm would not terminate. To prevent this situation, γ_i is increased by ϵ, also known as ϵ-complementary slackness. Basically, this allows assignments between persons and objects when the individual value differs from the maximum value by ϵ.

The set of persons having bid on object j is denoted as $P(j)$. During the assignment phase, each object j determines the person having submitted the highest bid as given by $i_j \in \text{argmax}_{i \in P(j)} \gamma_i$. The new price of the object is then set to $p_j + \gamma_{i_j}$.

In order to represent the ownership between a person and an object as well as the current price, a $N \times N$ matrix is

typically used. Generally, the algorithm provides an approximation, since it is only guaranteed to find the optimal solution if $\epsilon < 1/N$ and $a_{i,j} \in \mathbb{N}$ hold.

3 Related work

In this section, recent work from the field of sensor-based sorting and parallel strategies for solving the linear assignment problem is briefly reviewed.

3.1 Sensor-based sorting

Sensor-based sorting is typically used in the fields of food processing [8], waste management [1], and sorting of industrial minerals [9]. The process can be subdivided into the tasks of feeding, preparation, presentation, examination, discrimination, and physical separation [10, 11]. Preparation and presentation are realized by means of certain transport mechanisms, for instance conveyor belts, slides, or chutes. Usually, the goal is to unscramble the individual objects and achieve ideal flow control in terms of having all objects move at a defined velocity. During the examination, a task-specific sensor, possibly in combination with an appropriate illumination device, acquires the data [12, 13]. For the purpose of discrimination, data analysis is performed. For small, cohesive materials, physical separation is performed using an array of compressed air nozzles [14].

Conventional systems typically utilize scanning sensors. Recently, applying an area scan camera in place of a line scan camera has been proposed [2, 3]. The goal is to decrease the error in physical separation by selecting the nozzle(s) as well as the point in time to trigger the nozzle based on the results of a predictive tracking approach. In [15], it is shown that information derived from tracked objects can also be exploited for discrimination of products and hence be used to decrease the detection error. The challenge of respecting real-time requirements regarding multitarget tracking in sensor-based sorting is addressed in [16]. The authors propose a framework that dynamically selects an appropriate algorithm to solve the linear assignment problem based on the current system load. However, only a homogeneous hardware is assumed, and all algorithms are executed on a conventional CPU.

3.2 Parallel strategies for solving the linear assignment problem

Although this work focuses on the auction algorithm, it is worth mentioning that numerous algorithms for solving the linear assignment problem exist. An overview and comparison of some of the methods are provided in [5].

Due to its suitability for parallel processing, an implementation of the auction algorithm on a GPU is proposed in [6]. The authors present an implementation based on CUDA. They experimentally evaluate their implementation, comparing it with a sequential CPU implementation using a computer vision task, namely correspondence matching of 3D points. In [17], the authors present further improvements of the implementation. By splitting the $N \times N$ matrix into two unidimensional arrays, one holding the object prices and the other one the bids, they achieve lowering of memory usage. Results are presented in terms of memory usage, showing how especially problems involving huge datasets benefit from the approach.

An implementation of the auction algorithm on a field-programmable gate array (FPGA) is presented in [18]. The authors compare their experimental results with those presented in [6] and claim to achieve results ten times faster for certain problem sizes.

Parallel versions of two variants of the Hungarian algorithm, a different method for solving the linear assignment problem, are considered in [19]. The implementation is tailored to NVIDIA devices due to its implementation using CUDA. Also, the authors state that their implementation supports multi-GPU versions and consider up to 16 GPUs.

4 Multitarget tracking in sensor-based sorting

As has been mentioned, sensor-based sorting systems are typically designed according to a sorting task at hand. Materials to be sorted may strongly vary in terms of size, ranging from only a few millimeters, e.g., seeds, up to several centimeters, for instance minerals. For reasons of efficiency, the throughput of the system should generally be as high as possible while respecting quality requirements. Consequently, with respect to multitarget tracking, as many as thousands of objects may need to be tracked simultaneously. For systems using a conveyor belt for transportation, the applied belt speed may also vary. Typically, it is configured within a range of 1–5 ms^{-1}. Also, the image resolution needs to be sufficient for detecting the smallest possible characteristic that is of importance for material characterization. In many cases, it is only a fraction of a millimeter.

In many cases, the data retrieved by the sensor of a sorting system can be represented as an image, e.g., in optical sorting. Therefore, the measurements that serve as the input for the tracking algorithm need to be detected in the image data. This task is handled by various image

processing routines, such as filtering and segmentation, which are required during data evaluation in order to characterize the individual objects. The centroids of the extracted objects then represent a set of unlabeled measurements for a received image in our case.

In our system, multitarget tracking can be subdivided into four tasks, namely *state estimation*, *gating*, *association*, and *internal state management* [16]. For the purpose of *state estimation*, a standard Kalman filter is used. Assuming the assignments between the measurements and the tracks are given, we refine our knowledge about the positions of the particles by performing a Kalman filter update step for each individual particle. For the prediction step of the Kalman filter, we assume that a constant velocity model can be used to approximate the motion of the particles and state variables are given by the x and y of the measurement and the velocities in both directions, i.e., v_x and v_y. The prediction step has a complexity of $O(n)$, where,n denotes the number of current target tracks, which is expected to be (almost) equal to the number of measurements. The prediction yields the approximate position of the individual particles in the next time step, which then serve as the input for the *gating* and the *association* step at the next time step. The goal of *gating* is to partition the search space prior to association and hence split the problem into several smaller subproblems. This also allows for parallel processing during association. However, in this paper, gating is not considered. During *association*, the prediction of the existing tracks needs to be matched to the current measurements. This requires solving a linear assignment problem. Various algorithms exist, and they differ both in terms of computational complexity and whether they guarantee optimal results. In this paper, we focus on the auction algorithm for this task.

With reference to the description provided in Sect. 2, the sets N and M denote the predictions of the existing tracks and the measurements of the current frame. Initially, M contains the measurements (objects) and N the predictions (persons). The bidding kernel is implemented from the point of view of the persons, the assignment kernel from the point of view of the objects. Thus, the corresponding quantity indicates the problem size. However, due to the restriction that $|M| \geq |N|$ must hold true, the sets are possibly swapped. In our scenario, the distance between a measurement obtained and the prediction of a track is used to determine which measurement is to be associated with which track. The algorithm considered here utilizes a *cutoff* distance denoted as $d_{\max}$. Whenever the distance between the position $\underline{M}_i$ of measurement i and the position of the current prediction $\underline{N}_j$ of track j exceeds this distance, the utility is set to zero:

$$a_{i,j} = d_{\max} - \min\left\{d_{\max},\ d(\underline{M}_i, \underline{N}_j)\right\}. \tag{3}$$

This transformation is advantageous for the improvements discussed in Sect. 5.

Furthermore, an asymmetric problem is considered, i.e., it is not required that $|N| = |M|$ holds true. In our scenario, tracks for which no measurement yielded a utility greater than 0 are regarded as tracks to be erased. In the context of sensor-based sorting, this means that an object has left the observed area. However, due to possible occlusions, collisions, or poor object detection, tracks are not deleted immediately. Instead, a scoring system is applied. More precisely, each new track is assigned an initial score. Whenever a measurement is assigned to a track, the score is increased until it reaches a defined maximum score. Likewise, the score is decreased for frames in which no measurement was assigned to the track. When the score drops below zero, the track is finally deleted. Measurements that have not been assigned to any track are regarded as new tracks. These can be objects that just entered the observed area or measurements of tracks that have already been deleted. A more complex strategy for creation and deletion of tracks that takes the position of a measurement inside the observed area into account is presented in [4], but not considered in this paper.

5 Enhanced implementation of the auction algorithm

In this section, the proposed implementation of the auction algorithm for multitarget tracking is presented. It contains two improvements that increase the speed of the algorithm and/or lower memory usage. These improvements are integrated both in an OpenCL and CUDA implementation, for which results are presented in Sect. 6. It mainly consists of two kernels handling the bidding and assignment phase, respectively, and a kernel containing both phases as well as the convergence test. The latter is a necessity for the improvement proposed in Sect. 5.2.

5.1 Replacing the bidding matrix by one 1D array

As has been discussed in Sect. 3, an approach replacing the $N \times M$ matrix by two arrays has been proposed in [17]. Although it is successfully demonstrated how the approach significantly lowers memory usage, no results on the impact on processing time are presented. Also, the method suffers from requiring synchronization whenever updating the current owner and price of an object. This is due to possible race conditions whenever multiple persons are bidding on one and the same object concurrently. Therefore, we propose to reduce the data structure to one array only. This is made possible by encoding the bid as well as

the person in a single value. More precisely, the higher order bits of a value are used to store the current bid, while the lower bits contain the person that placed the bid. Context-specific fixed-point numbers are then a convenient way of encoding the bid. An example considering a 32-bit data type is provided in Fig. 4. This approach allows the utilization of atomic functions for updating a value. For instance, an atomic max function can be used to update the value if a higher bid is posed. If two persons submit the same bid, the person with the greater *Person ID* wins.

The approach is further designed such that it does not yield any relevant restrictions regarding the highest possible bid. This is achieved by determining a factor K in correspondence with $d_{\max}$. In order to do so, the upper bound of the possible bid increment can be calculated such that

$$\begin{aligned} &\gamma_i \leq \gamma_{\max} \quad \text{with} \\ &\gamma_{\max} = d_{\max} + \epsilon, \quad i \in [1, \ldots, N] \end{aligned} \tag{4}$$

holds true. For a scaled bid c, Eq. (5) holds, where K indicates the scaling factor. K is chosen such that the number of available bits suffices for c for a given $d_{\max}$ and ϵ.

$$0 \leq c \leq K \cdot \gamma_{\max} = K(d_{\max} + \epsilon) \tag{5}$$

However, the presented approach implies a restriction regarding the number of persons that can be considered. For cases where high numbers are required, larger data types can be used, for instance 64-bit instead of 32-bit.

This approach is advantageous for several reasons. Firstly, memory usage and therefore the amount of data that potentially requires transfer to GPU memory is significantly reduced and can be formulated as follows. In cases when $|N| \approx |M|$, the complete bidding matrix consists of $|N|^2$ entries. Applying the improvements from [17], the number of entries can be decreased to $2|N|$. The improvement proposed in this paper reduces the amount of required memory to $|N|$. This is particularly advantageous for embedded systems, which are often used in sensor-based sorting. Furthermore, compared to [17], the proposed approach allows utilization of atomic functions instead of locking two fields whenever an ownership is to be updated. More precisely, using a mutex instead, at least one atomic function would be necessary to receive the lock, followed by at least two read/write operations, and another atomic call to release the lock. In our case, only one atomic function call is necessary. Also, fewer fields require resetting, i.e., setting to zero, between the iterations. Another advantage lies in avoiding inefficient access of memory. The proposed approach enables memory coalescing, which is not possible when using a bidding matrix, because either the bidding or the assignment kernel (depending on whether the sets were swapped) requires all values from a column of the matrix to be accessed in one time step. These elements are then not located consecutively in memory. Lastly, identifying the highest bid during the assignment phase is not required anymore, since the information is directly stored in the corresponding field.

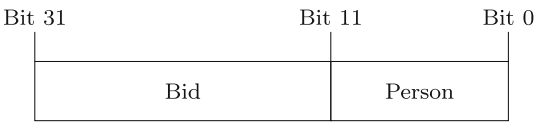


Fig. 4 Example of a partitioning of a value stored in the bidding array. In this case, a 32-bit data type is assumed. The 20 highest bits are used to store the current bid and the 12 lowest for the person

5.2 Synchronization on the GPU

An implementation of the auction algorithm requires the bidding phase to be completed for all persons before the assignment phase may start. Likewise, the assignment phase must be completed before the convergence test can be run. Consequently, this requires synchronization steps between each of the phases.

OpenCL enables synchronization of *work-items* which are part of the same *work-group* on-GPU. We propose to exploit this property, such that no synchronization handled by the CPU is required. However, due to the restriction that all *work-items* need to be part of the same *work-group*, the extent to which this improvement can be used depends on the hardware. Yet, it is important to note that this information can be retrieved during run time and hence whether to enable this feature or not can be dynamically decided for each time step. Likewise, CUDA supports the concept of several *threads*, which may be part of the same *block*. Aforementioned synchronization procedures are realized in the same way.

6 Test methodology and experimental results

In this section, the environment used for experimentation and corresponding results are presented.

6.1 Setup and datasets

Results presented below were obtained using six datasets. Four of these were generated using a particle-based simulation of an optical belt sorter as described in [20]. Here, the discrete element method (DEM) was employed to accurately model the particle–particle and particle–wall interactions within the sorting system. The DEM was first introduced in 1979 by Cundall and Strack [21] and has since found application in various engineering tasks

concerned with granular materials. To calculate the precise translational and rotational motion of every particle at a predefined time step, Newton and Euler's equations of motion are used:

$$m_i \frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{F_i^c} + \mathbf{F_i^g}, \tag{6}$$

$$\hat{I}_i \frac{d\mathbf{W}_i}{dt} + \mathbf{W}_i \times (\hat{I}_i \mathbf{W}_i) = \Lambda_i^{-1} \mathbf{M}_i, \tag{7}$$

In the translational equation, the particle mass m_i multiplied with the particle acceleration $d^2\mathbf{x}_i/dt^2$ equals the sum of the contact force $\mathbf{F}_i^c$ and the gravitational force $\mathbf{F}_i^g$. The second equation gives the angular acceleration $d\mathbf{W}_i/dt$ as a function of the angular velocity $\mathbf{W}_i$, the external moment resulting out of the contact forces $\mathbf{M}_i$, the inertia tensor along the principal axis $\hat{I}_i$, and the rotation matrix converting a vector from the inertial into the body fixed frame Λ_i^{-1}. The contact forces between particles and between particles and walls are separated into a normal and a tangential component. The normal component is calculated with a linear spring damper model and the tangential contact force with a linear spring limited by the Coulomb condition. In addition, the rolling friction of the spherical particles is also included in the resulting external moment.

For the datasets obtained using the DEM simulation, the simulated conveyor belt runs at 1.5 ms^{-1}. The remaining two datasets were recorded on an actual sorting system with a conveyor belt speed of approximately 2.7 ms^{-1} and a resolution of approximately 0.3 mm per pixel. A more detailed description of all datasets is provided in Table 1. The name of the dataset describes the type of objects that were applied on the sorting system. For the simulated datasets, detailed information regarding contact forces, etc., is provided in [20]. As can be seen, they strongly differ in terms of included measurements overall as well as per frame. Additionally, it is noted that for dataset *Spheres 1*, spheres with a diameter of 5 mm were simulated, while for *Spheres 2*, the diameter was reduced to 2.5 mm in order to simulate an even higher throughput in terms of the number of objects. This eventually leads to varying problem sizes considered in the linear assignment problem.

Generally, the parameters for the scoring system as described in Sect. 4 need to be set carefully. However, with respect to the datasets obtained via simulation, it is important to note that the data is noise free. Therefore, occlusions and missing measurements do not occur. Although collisions might occur between the objects, the input data for the tracking system is already reduced to the centroids of the objects and two centroids can never occupy the same point in space. Below, an initial track score of 5, an increase of 2, decrease of 1, and maximum score of 10 are considered without further variation.

All experiments presented were run on an Intel Core i7-6700 with 16 GB DDR-4-2133 RAM. The operating system was Microsoft Windows 10. The CUDA code was run on a dedicated graphics card, namely NVIDIA Titan X with Pascal architecture. CUDA compute capability version 3.5 was used. With respect to the OpenCL implementation, results are presented for both the aforementioned GPU and an integrated one, namely an Intel HD 530. For OpenCL, version 1.2 was used. The maximum work-group size for the Titan X GPU is 1024, and for the Intel HD 530 GPU, it is 256. For all experiments, the maximum possible work-group size was chosen, i.e., either the problem size or the maximum size supported by the hardware. It is important to note that for the improvement discussed in Sect. 5.2 to apply, it is a necessity that the problem size does not exceed the possible work-group size.

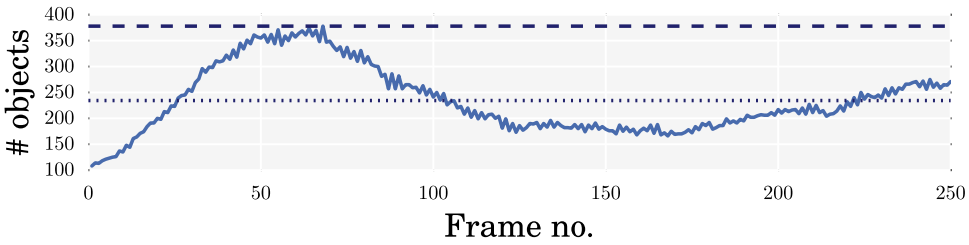
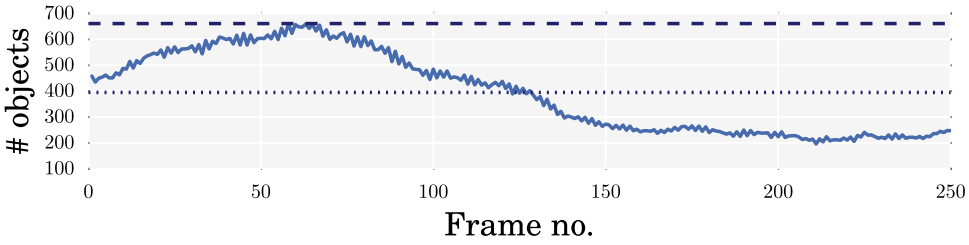
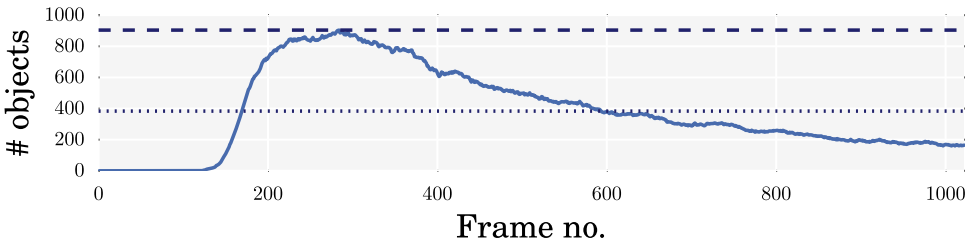
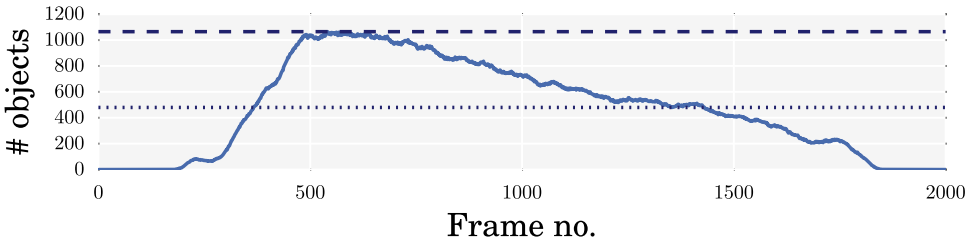
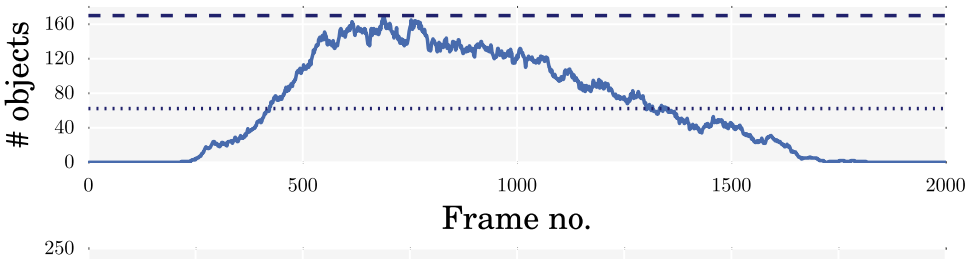
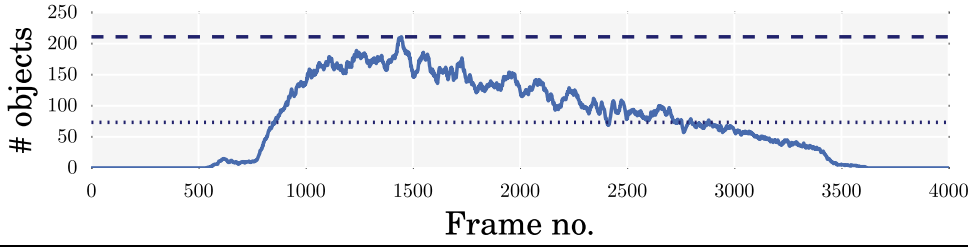
In addition to absolute times reported, speedup values are used for comparison with a reference in the remainder. Values reported are defined as $t_{\text{reference}}/t_{\text{proposed}}$. As a reference, the implementation published with [6] is used. For the CUDA-based code, the original sources are used, and for OpenCL a porting of the sources.

6.2 Experimental results

The improvements of the auction algorithm discussed in Sect. 5 were implemented both in the CUDA framework published with [6] and a corresponding OpenCL implementation. The results are presented in Fig. 5. In general, as one might assume, the data show that the fastest processing is achieved using CUDA on the Titan X graphics card and the slowest using OpenCL in combination with the integrated HD 530 GPU. Regarding the quality, it is stated that the implementation under evaluation results in equal associations as the one presented in [6]. Quantitative results regarding the quality for datasets that were generated using DEM simulation are provided in Table 2. As with the definition presented in [16], we regard an error to have occurred whenever measurements of the same object at different time points have been assigned to different tracks. This definition of the error respects both the case that a track of an actual object was interrupted, for instance by missing measurements, and that an object was assigned to a wrong track. In Table 2, we present the total number of objects in the dataset for which an error occurred. The number of interruptions or false assignments per object is not considered.

As can be seen from Fig. 5, irrespective of the framework and the hardware, a significant reduction in time required can be achieved by the proposed improvements. Although the absolute time required differs strongly, it can

Table 1 Summary of the datasets considered for the experimental evaluation

Name	Source	Sample rate (Hz)	No. of objects per frame
Pepper corn 1	Camera	220	
Pepper corn 2	Camera	220	
Spheres 1	Simulation	100	
Spheres 2	Simulation	100	
Spheres and plates	Simulation	100	
Cylinders	Simulation	200	

The dotted lines indicate the average number of objects to be tracked per frame and the dashed lines the maximum number of objects to be tracked in a single frame

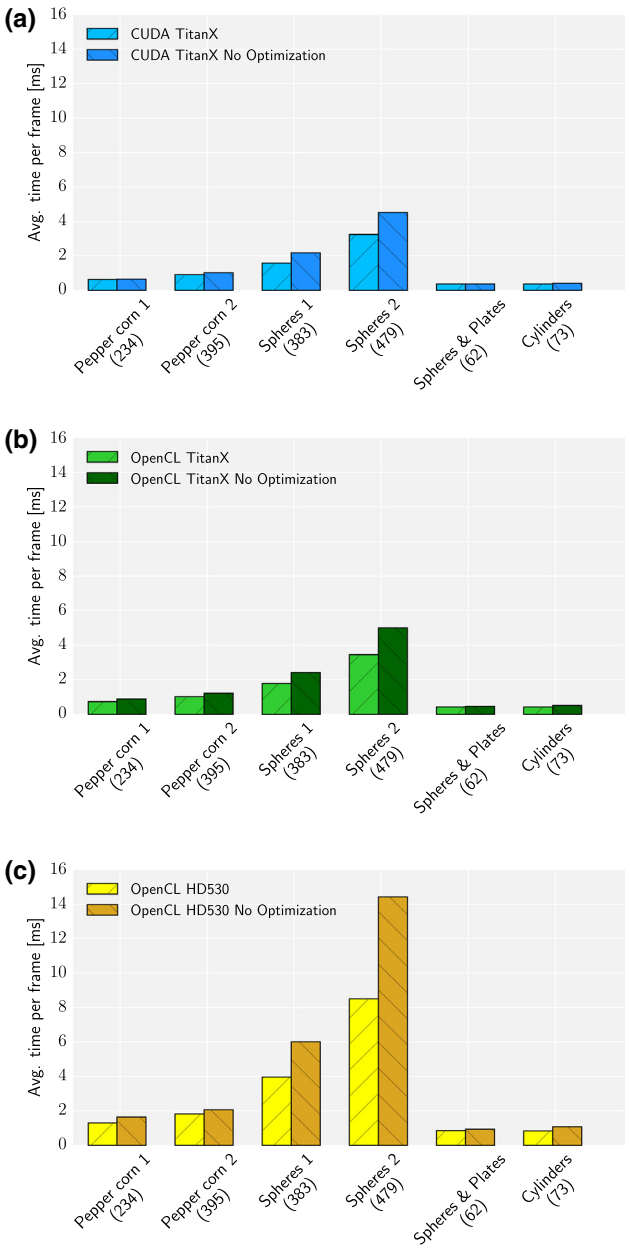


Fig. 5 Processing times for the different platforms and hardware components for the different datasets. The number in brackets for each dataset shows the average number of objects to be tracked per frame. **a** Processing times using the CUDA framework and the Titan X dedicated graphics card. **b** Processing times using the OpenCL framework and the Titan X dedicated graphics card. **c** Processing times using the OpenCL framework and the integrated HD 530 graphics unit

be stated that the decrease in the percentage is comparable between the different setups. In all cases, our approach outperforms the conventional implementation. A quantitative comparison in terms of speedups is provided in Table 3. For datasets with a comparatively low number of

Table 2 Overview of the tracking quality for the datasets obtained from simulation

Dataset	Total objects	Errors
Spheres 1	12134	130
Spheres 2	29693	19
Spheres and plates	3599	5
Cylinders	4412	0

Table 3 Speedup values compared with an implementation without the proposed optimization

Dataset	CUDA Titan X	OpenCL	OpenCL HD 530
Pepper corn 1	1.04	1.21	1.27
Pepper corn 2	1.10	1.20	1.13
Spheres 1	1.38	1.36	1.52
Spheres 2	1.40	1.45	1.70
Spheres & Plates	1.04	1.09	1.11
Cylinders	1.12	1.20	1.28

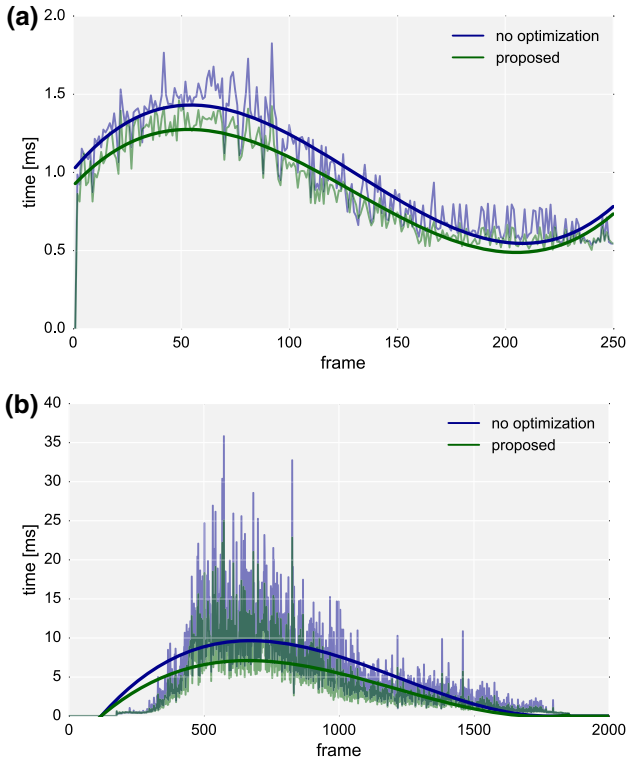


Fig. 6 Time per frame required by the implementation based on CUDA with and without the proposed improvements. Additionally to the exact measurements, a fitted trend curve is shown. **a** Dataset *Pepper corn 2* **b** Dataset *Spheres 2*

Table 4 Ratio of the average required and the available processing time for the auction algorithm considering a camera operating at 200 Hz

Dataset	CUDA Titan X		OpenCL Titan X		OpenCL HD 530	
	No optimization (%)	Proposed (%)	No optimization (%)	Proposed (%)	No optimization (%)	Proposed (%)
Pepper corn 1	13	13	18	15	33	26
Pepper corn 2	20	18	20	20	41	36
Spheres 1	43	32	48	36	120	79
Spheres 2	91	65	100	69	288	170
Spheres and plates	7	7	9	9	10	8
Cylinders	8	7	10	8	21	17

For each dataset, best results are achieved using CUDA and the Titan X GPU

average measurements per frame, such as *Spheres & Plates*, rather low speedups ranging from 1.04 for the CUDA-based implementation up to 1.11 utilizing OpenCL and the HD 530 graphics chip are obtained. However, for the dataset including the highest number of average measurements, namely *Spheres 2*, a speedup of 1.4 is reported for CUDA on the Titan X, 1.45 for OpenCL on the Titan X, and even 1.7 for OpenCL on the HD 530. Also, from the description of the datasets as provided in Table 1, it can be observed that especially for comparatively large problem sizes, our improvements are fully effective. The latter can further be observed from Fig. 6. Here, two examples of the required time per frame are provided.

Solving the linear assignment problem is one of many steps that must be performed during data analysis in sensor-based sorting. Additional steps include pre-processing of the image, detecting regions containing objects, and classification of those objects, which can be implemented exploiting pipeline parallelism. Therefore, it is of particular interest how much of the available time, which is defined in terms of the camera acquisition rate, is consumed by solving the linear assignment problem. In this way, it can be stated whether real-time requirements can be fulfilled by the system. Below, we assume a camera operating at 200 Hz and hence 5 ms of available processing time per frame. The ratio of time required by the auction algorithm for this scenario, both conventional and including the proposed improvements, is provided in Table 4. Here, it can also be seen that the time required can be reduced significantly for datasets causing high work load, such as both *Spheres* datasets. Considering the *Spheres 2* dataset, the ratio is reduced by 26 percentage points in the CUDA implementation running on the Titan X graphics card and by as much as 31 percentage points using OpenCL on the same hardware. With respect to the HD 530 hardware, it becomes clear that large problem sizes cannot be handled under the given time constraint. Further, it is important to note that these numbers are based on the average processing time per frame for the individual datasets. From Fig. 6, it becomes clear that not exceeding 5 ms of processing time is not possible for each individual frame.

7 Conclusion

In this paper, improvements to a GPU-based implementation of the auction algorithm were proposed that result in lower memory usage as well as increased speed. Regarding the latter, it was demonstrated that the approach outperforms conventional implementations of the algorithm. In the best case, it performs 1.7 times as fast and the geometric average of the speedup is 1.24 when averaged over all platforms. With respect to the different hardware considered, the geometric average of the speedup using CUDA is 1.17 and 1.25 for OpenCL when run on the Titan X GPU and even 1.32 using OpenCL in combination with the HD 530 GPU. Also, it was shown that, especially for huge problem sizes, the proposed approach can support fulfilling firm real-time requirements. This was further elaborated in the example of data analysis in sensor-based sorting.

Regarding future work, the aim is to focus on particularly challenging situations in terms of computational burden. The experimental results presented reveal that although the run time can be significantly reduced and acceptable processing times can be achieved on average, the real-time requirement cannot be fulfilled for certain individual frames. This problem may be tackled by dynamically adapting ϵ such that it becomes more likely that fewer auction iterations are required. Also, a hard threshold regarding the maximum number of allowed iterations may be introduced. However, it is important to note that the output quality of the auction algorithm does not increase monotonically over the iterations. Therefore, the approach would result in a loss of quality, which is not the case for the approach presented in this paper.

Acknowledgements IGF project 18798 N of research association Forschungs-Gesellschaft Verfahrens-Technik e.V. (GVT) was supported by the AiF under a program for promoting the Industrial Community Research and Development (IGF) by the Federal Ministry for Economic Affairs and Energy on the basis of a resolution of the German Bundestag.

References

1. Kępys, W.: Opto-pneumatic separators in waste management. In: żynieria Mineralna **17** (2016)
2. Pfaff, F., Baum, M., Noack, B., Hanebeck, U.D., Gruna, R., Längle, T., Beyerer, J.: Tracksort.: Predictive tracking for sorting uncooperative bulk materials. In: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), IEEE, pp. 7–12 (2015)
3. Pfaff, F., Pieper, C., Maier, G., Noack, B., Kruggel-Emden, H., Gruna, R., Hanebeck, U.D., Wirtz, S., Scherer, V., Längle, T., et al.: Improving optical sorting of bulk materials using sophisticated motion models. tm-Tech. Mess. **83**(2), 77–84 (2016)
4. Pfaff, F., Pieper, C., Maier, G., Noack, B., Kruggel-Emden, H., Gruna, R., Hanebeck, U.D., Wirtz, S., Scherer, V., Längle, T., Beyerer, J.: Simulation-based evaluation of predictive tracking for sorting bulk materials. In: 2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI). (Sept 2016) pp. 511–516
5. Dell'Amico, M., Toth, P.: Algorithms and codes for dense assignment problems: the state of the art. Discret. Appl. Math. **100**(1–2), 17–48 (2000)
6. Vasconcelos, C.N., Rosenhahn, B.: Bipartite graph matching computation on GPU. In: International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, Springer, pp. 42–55 (2009)
7. Bertsekas, D.P.: The auction algorithm: a distributed relaxation method for the assignment problem. Ann. Oper. Res. **14**(1), 105–123 (1988)
8. Narendra, V., Hareesha, K.: Prospects of computer vision automated grading and sorting systems in agricultural and food products for quality evaluation. Int. J. Comput. Appl. **1**(4), 1–9 (2010)
9. Lessard, J., de Bakker, J., McHugh, L.: Development of ore sorting and its impact on mineral processing economics. Miner. Eng. **65**, 88–97 (2014)
10. Kleiv, R.A.: Pre-sorting of asymmetric feeds using collective particle ejection. Physicochem. Probl. Miner. Process. **48**(1), 29–38 (2012)
11. Batchelor, A., Ferrari-John, R., Katrib, J., Udoudo, O., Jones, D., Dodds, C., Kingman, S.: Pilot scale microwave sorting of porphyry copper ores: part 1-Laboratory investigations. Miner. Eng. **98**, 303–327 (2016)
12. Cubero, S., Aleixos, N., Moltó, E., Gómez-Sanchis, J., Blasco, J.: Advances in machine vision applications for automatic inspection and quality evaluation of fruits and vegetables. Food and Bioprocess Technol. **4**(4), 487–504 (2011)
13. Gruna, R., Beyerer, J.: Feature-specific illumination patterns for automated visual inspection. In: Proceedings IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Graz, Austria (May 2012)
14. Ferreira, T., Sesmat, S., Bideaux, E., Sixdenier, F.: Experimental analysis of air jets for sorting applications. In: 8th FPNI Ph.D. Symposium on Fluid Power, American Society of Mechanical Engineers (2014) V001T01A007–V001T01A007
15. Maier, G., Pfaff, F., Pieper, C., Gruna, R., Noack, B., Kruggel-Emden, H., Längle, T., Hanebeck, U.D., Wirtz, S., Scherer, Viktor Beyerer, J.: Improving material characterization in sensor-based sorting by utilizing motion information. In: OCM 2017 - Optical Characterization of Materials, KIT Scientific Publishing (2017, in press)
16. Maier, G., Pfaff, F., Pieper, C., Gruna, R., Noack, B., Kruggel-Emden, H., Längle, T., Hanebeck, U.D., Wirtz, S., Scherer, V., Beyerer, J.: Fast multitarget tracking via strategy switching for sensor-based sorting. In: 2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), pp. 505–510 (Sept 2016)
17. Vasconcelos, C.N., Rosenhahn, B.: Bipartite graph matching on GPU over complete or local grid neighborhoods. In: International Work-Conference on Artificial Neural Networks, Springer, pp. 425–432 (2011)
18. Zhu, P., Zhang, C., Li, H., Cheung, R.C., Hu, B.: An FPGA-based acceleration platform for auction algorithm. In: IEEE International Symposium on Circuits and Systems. IEEE **2012**, pp. 1002–1005 (2012)
19. Date, K., Nagi, R.: GPU-accelerated Hungarian algorithms for the Linear Assignment Problem. Parallel Comput. **57**, 52–72 (2016)
20. Pieper, C., Maier, G., Pfaff, F., Kruggel-Emden, H., Wirtz, S., Gruna, R., Noack, B., Scherer, V., Längle, T., Beyerer, J., et al.: Numerical modeling of an automated optical belt sorter using the discrete element method. Powder Technol. **301**, 805–814 (2016)
21. Cundall, P.A., Strack, O.D.: A discrete numerical model for granular assemblies. Geotechnique **29**(1), 47–65 (1979)

Georg Maier is with the Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation IOSB, Karlsruhe, Germany. His research interests include different aspects of image processing, in particular algorithmic aspects, with a focus on real-time capabilities.

Florian Pfaff is a research assistant at the Intelligent Sensor-Actuator-Systems Laboratory at the Karlsruhe Institute of Technology. His research interests include a variety of estimation problems such as nonlinear filtering, multitarget tracking, and estimation in the presence of both stochastic and non-stochastic uncertainties.

Matthias Wagner received the M.Sc. degree in Computer Science from the Karlsruhe Institute of Technology in 2016. During the final phase of his studies, he was involved in multitarget tracking for sorting of bulk goods as a research assistant at the Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation IOSB, Karlsruhe, Germany. His research interests include parallel programming with a focus on graphics processing units.

Christoph Pieper is a research assistant at the Department of Energy Plant Technology at the Ruhr University Bochum. His research interests include numerical simulation of fluidized particle systems with the discrete element method (DEM) and computational fluid dynamics (CFD).

Robin Gruna is research group manager at the Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation IOSB in Karlsruhe. His research interests include hyperspectral imaging, machine vision, computational illumination, and chemometrics.

Benjamin Noack received his diploma in computer science from the Karlsruhe Institute of Technology (KIT), Germany, in 2009. Afterward, he obtained his Ph.D. in 2013, at the Intelligent Sensor-Actuator-Systems Laboratory, Karlsruhe Institute of Technology (KIT), Germany. Since 2013, he is a senior researcher at the Karlsruhe Institute of Technology (KIT), Germany. His research interests are in the areas of multi-sensor data fusion, distributed and decentralized Kalman filtering, combined stochastic and set-membership approaches to state estimation, and event-based systems.

Harald Kruggel-Emden is professor and head of the department of Mechanical Process Engineering and Solids Processing at the Technical University of Berlin. His research interests include discrete element modeling with coupled fluid flow, material preparation and drying technology, bulk solids handling, and chemical looping combustion.

Thomas Längle is adjunct professor at the Karlsruhe Institute of Technology (KIT), Karlsruhe, and the head of the business unit "Vision Based Inspection Systems" (SPR) at the Fraunhofer IOSB in Karlsruhe, Germany. His research interests include different aspects of image processing and real-time algorithms for inspection systems.

Uwe D. Hanebeck is a chaired professor of Computer Science at the Karlsruhe Institute of Technology (KIT) in Germany and director of the Intelligent Sensor-Actuator-Systems Laboratory (ISAS). He obtained his Ph.D. degree in 1997 and his habilitation degree in 2003, both in Electrical Engineering from the Technical University in Munich, Germany. His research interests are in the areas of information fusion, nonlinear state estimation, stochastic modeling, system identification, and control with a strong emphasis on theory-driven approaches based on stochastic system theory and uncertainty models. He is author and coauthor of more than 400 publications in various high-ranking journals and conferences and an IEEE Fellow.

Siegmar Wirtz is the deputy head of the Department of Energy Plant Technology at the Ruhr University Bochum. His research interests include numerical simulation of reactive gas-solid flows, extension of commercial CFD-codes, discrete element modeling with coupled fluid flow.

Viktor Scherer is the head of the Department of Energy Plant Technology at the Ruhr University Bochum. His research interests include energetic conversion of fossil fuels and biomass as well as related industrial applications and experimental and theoretical investigation of energy and high-temperature processes.

Jürgen Beyerer is full professor for informatics at the Institute for Anthropomatics and Robotics at the Karlsruhe Institute of Technology (KIT) since March 2004 and director of the Fraunhofer Institute of Optronics, System Technologies, and Image Exploitation IOSB in Ettlingen, Karlsruhe, Ilmenau, and Lemgo. Research interests include automated visual inspection, signal and image processing, variable image acquisition and processing, active vision, metrology, information theory, fusion of data and information from heterogeneous sources, system theory, autonomous systems, and automation.