GPU-related efficient visual information processing approaches

Theses of the PhD dissertation

Anna Gelencsér-Horváth

Scientific advisors:

György Cserey, PhD

Pázmány Péter Catholic University

Kristóf Karacs, PhD

Pázmány Péter Catholic University

Pázmány Péter Catholic University

Faculty of Information Technology and Bionics

Roska Tamás Doctoral School of Sciences and Technology

Budapest, 2023

1 Introduction

Since visual perception provides us with a significant amount of information about our surroundings, processing visual information to mimic human vision with computers is a widely researched area with many motivations. The speed of computer vision algorithms is critical in many applications including those that require real-time information processing, such as a navigation system. Particle filters can be traced back to at least the 1990s [1] and they correspond to a robust approach to deal with nonlinear state-space models subject to additive noise, not restricted to Gaussian noise [2]. Particle filters (PF) are both part of the SMCM algorithm family and can be considered an extension of the Kalman filter [3]. The potential use of particle filtering goes far beyond predicting time series, such as in financial mathematics [4, 5] or position tracking, despite the challenge of applying particle filtering to high-dimensional systems [6]. It is used in many approaches where the input is an image, or a series of images, including image reconstruction [7], object detection [8], navigation [9], segmentation [10, 11], contour detection [12], and in tracking with occlusion [13].

A benchmark time series and its observation [2] is illustrated in Figure 1.

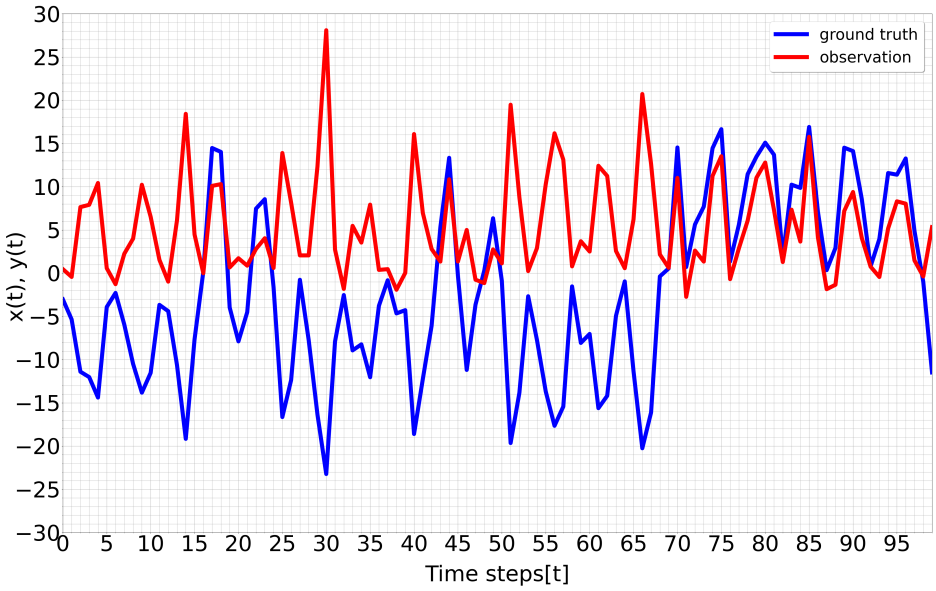

Figure 1: A time series (ground truth with blue) and its observation (red) of a particle filter benchmark model [2].

The PF algorithm is computationally expensive due to the resampling step according to the *complete* cumulative distribution [14]. The efficient and high prediction quality implementation to parallel architectures has been widely researched [15–24].

Graphics processing units (GPUs) represent an attractive implementation platform as they have high computational efficiency, while the price of a device is relatively low. This increased capacity can only be well exploited if algorithms are adapted to the characteristics of the hardware architecture being used. Nevertheless, the method used for resampling has a high impact on the prediction quality. As a comparison shows in [25], different resampling strategies offer trade-offs between speed and prediction quality. Therefore, an adequate parallel implementation of PF that retains local connections and the information exchange among the particles during resampling can achieve a remarkable speed-up with no degradation of the estimation quality. During my research, I addressed the acceleration of the particle filtering algorithm, to re-design it to fit GPU architectures of the time the most efficiently [A1, A3].

My research goal was to introduce a fast particle filter with sequential importance resampling (SIR) [26] and provide a novel method that allows the implementation on GPUs to retain high-quality prediction. Cellular particle filter(CPF) [27] introduces a promising approach for the resampling problem by changing the logic representation of the PF to a two-dimensional (2D), locally connected grid inspired by the cellular neural network (CNN) architecture [28]. Due to the CNN-type representation and the decreased dimension of resampling sets, CPF offers a solution to the problem of reduced local information change, which is stated in [17]. Although the prediction is of the same quality as for sequential implementation, this representation is not optimal in terms of exploiting GPU architecture to achieve high efficiency. Hence, I intended to re-design the algorithm to achieve an efficient implementation that fits the characteristics of the GPUs and thus, can exploit and computational capability and speed of GPUs.

In addition to the importance of increasing the efficiency of the core algorithms, what really matters in engineering tasks is the total time required for executing a process, including data acquisition and preparation. In addition to optimizing for the speed of individual components, such as a particle filter used for tracking, we also need to consider the need for suitable input usually provided by segmentation. Forming regions by organizing and labeling each pixel in the image into groups that belong to the same object, segmentation is a strategy for understanding and processing visual information. Therefore, I focused on segmentation and contour detection, as it is challenging for region-based methods based on pixel similarity to avoid merging similar adjacent objects or parts thereof. In addition to considering region similarities based on colors on an over-segmented image, created with a segmentation based on Mean Shift [29], I intended to exploit contour detection based on basic geometrical properties of the segments [A4, A5].

However, in addition to image processing and machine learning methods, over the past decade, there has been a rapid development of deep learning-based approaches for segmentation [30]. After a break between 2015 and 2021, I began investigating the segmentation of neighboring, highly similar areas using deep neural networks. The research was motivated by surveillance videos of unmarked rats, where even state-of-the-art algorithms failed in multi-animal tracking without a single ID switch due to heavy occlusions. Deep networks are effective tools for retrieving visual information automatically, and if sufficient training data is available, the final model can learn a high-dimensional representation of important image features - similar to the human brain. Compared to traditional algorithms, deep networks require a task-dependent amount of labeled data for training. Therefore, we can consider the quality of prediction and the time spent on data acquisition and annotation as a trade-off. For complex scenes, the collection of training data may be beyond the possibilities in terms of computational or human resources. Even if the required amount of data of adequate type is available, annotation still needs considerable time. Therefore, in terms of efficiency, we need to consider not only the quality and speed of the prediction, but also the time required to collect, prepare, and annotate the data for training. In the case of so-called end-to-end networks [31, 32], the training of a single, even complex structure provides the final prediction for the raw, not pre-processed input data. However, defining such a network can be a significant challenge, as it depends on the specific task and the characteristics of the input images. Composite AI [33, 34] is an approach that can help overcome limitations of end-to-end deep networks. For a complex task, deep learning, machine learning, image processing, higher-level logic, and even domain-specific knowledge can be used to create a pipeline that ensures a high-quality performance for challenging scenes with multiple unknown objects, occlusions, noises, and similarities, among others. Training the network on synthetic data, where the ground truth labeling is available by construction, saves considerable time and effort. However, the distribution and image characteristics of the synthetic training data must be sufficiently close according to an appropriate metric to those of the candidate inputs to ensure high-quality prediction and to avoid overfitting to artifacts, which is often highly challenging.

My goal was to create a method for high-quality segmentation of highly similar instances with unsupervised trained deep learning based on synthetically generated data that is automatically annotated to reduce the need for human effort. My research was motivated by surveillance videos of unmarked rats, where even state-of-the-art algorithms failed in multi-animal tracking without a single ID switch due to heavy occlusions.

I targeted a combination algorithm of a region-based detection approach [35] and contour detection that exploits inner boundaries of occluding instances, which provides a high-quality segmentation and allows reliable id-tracking using a propagation approach presented in [36]. I addressed the training of deep networks for edge detection based on synthetic data: a dataset is generated fully automatically (based on only frames with non-occluding instances), without any human annotation to train an edge detection network for detecting the separating boundary between occluding identical objects with a static background. A sample image pair is shown in Figure 2.

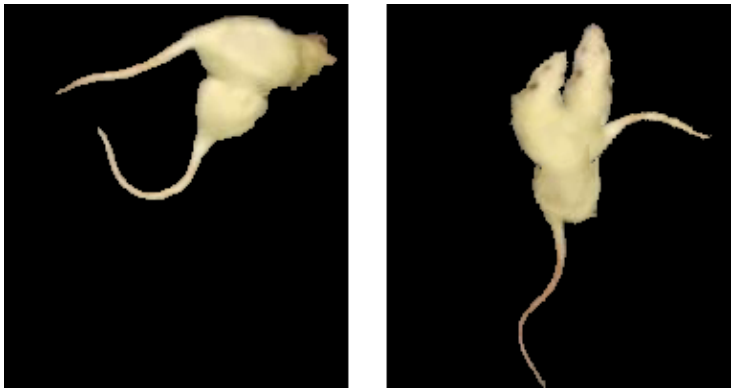

Figure 2: A real frame (on the left) and a synthetic frame (on the right) of two overlapping, highly similar, unmarked rats after background removal. The similar characteristics of a region of a single instance and those of the area created by the two overlapping rats pose a challenge in recognizing instance boundaries with traditional edge detection algorithms. Deep networks provide promising edge detection results and automatically annotated images that are similar to the real frames can significantly reduce the annotation required for training to facilitate a large number of biological studies, which otherwise would not be feasible.

As the "continuity" of the inner boundary edges required further improvement for precise segmentation, I was searching for a generative method, to provide an edge "completion" suited to the detected edge characteristics.

2 Methods of Investigation

To address the question of adapting a Particle Filter to GPU I relied on the available literature on parallel, distributed and local particle filters [15–24, 27] during my research in 2013, considering the following major aspects. First, information share ratio among the particles to minimize degradation of the quality of estimation compared to the original algorithm, which resamples according to the complete cumulative distribution.

Second, characteristics of the different GPU memory types, to improve kernel running times and on-device approaches for synchronization and random number generation to minimize the time consuming transfers between CPU and GPU. Although, NVIDIA Mersenne Twister included in the CUDA SDK seems to offer a promising solution, the distribution of the random numbers proved to be insufficient for low amount (hundreds and even thousands) of numbers. Therefore, I explored possible solutions and finally proposed two different approaches for random number generation.

To compare the quality of the estimation and the running time I used two benchmark models applied for evaluation in several state-of-the-art articles, available by the time of the research. On the one hand, the non-autonomous, non-linear model with a continuous state space used in [2, 37–39], and on the other hand, a bearings-only tracking model presented in [40] and used in [22, 23]. For evaluation, I used an NVIDIA GeForce GTX 550 Ti GPU with 1-GB GDDR memory, compute capability 2.1, and CUDA toolkit 4.1 with driver version 295.49.

To improve the quality of tracking highly similar occluding instances with keeping identity labels based on instance segmentation I considered the use of composite AI techniques as described in the Chapter 1. If different approaches as sub-modules are combined in a pipeline, the accuracy and performance can be better than that of an end-to-end approach [31, 32], moreover, each sub-module can be developed and optimized independently, and the results from one sub-module can be used to guide the development of the next sub-module. To improve instance segmentation the complete visual context (such as constraints) can be incorporated through higher logic into appropriately designed pipeline, while machine learning algorithms and computer vision techniques are used together. Combining two algorithms that can be considered as dual approaches [41], such as region detection and edge detection, allows for exploiting the complementary information that helps to mitigate the limitations of individual methods.

Based on the available literature deep learning methods are more promising for complex scenes than traditional edge detection techniques. Therefore I relied on the state-of-the art deep learning approaches presented in the literature [42–45] to train an edge detection deep pipeline. To train the networks I considered the prediction quality and the time required for annotating the training data. To avoid the monotone and time consuming human annotation, I considered the method introduced in [36] and explored the required transformations to provide a similar characteristic to real, non-augmented test frames. The dataset is derived from a 20-minute, 25 fps observation video with a resolution of 1280x720, which was provided by the Department of Physiology and Neurobiology of the Eötvös Loránd University (ELTE). The frames used as image inputs

are the same as those in the article [36], which were extracted and cropped to the region of the box, resulting in a final image resolution of 640x420 saved in *png* format. I synthetically constructed the training data from the first 10 minutes of the video containing 15 000 frames, as described in Thesis I.1 and Thesis 1.2. For training the Edge Detection [42] and Edge Completion [44] networks I used all 9 233 frames with non-occluding instances by determining the number of foreground objects based on histogram intensity and a background image constructed as a mode of 2 000 frames. For training the CycleGAN I selected 2 200 frames randomly from the ones with occluding instances. I chose the number of the training images for CycleGAN to be of a similar order of magnitude as the training dataset shown in the CycleGAN work, and based on the results, I considered it to be indeed appropriate.

To evaluate the quality of the segmentation results of the pipeline, 18 sequences of 200 images were selected by observation with challenging occlusions for human annotation. In this set 1 669 frames contain non-separated instances. See the number of training and test images for the different deep networks of the pipeline in Table 1. My goal was to minimize human annotation and attention time, therefore, time measurements are only indicative. 15 epochs for DexiNed took 6 hours, 10 epochs for CycleGAN took 1 hour, 20 epochs for EdgeConnect took 16 hours[1].

Table 1: Number of images for training and testing the different deep networks of the pipeline. 90% of the training images were used as an actual training set, and 10% as a validation set.

	Number of images			
	Train			Test
	RGB/aRGB	aGT(edge)	Non-closed contour	
DexiNed	92 330	92 330		3 600
CycleGAN		2 200	2 200	200
EdgeConnect	92 330	92 330	92 3300	3 600

The annotation was split between four annotators and each annotator's segmentation was validated by another annotator.

I extended the proposed pipeline with the tracking approach presented in [35] and compared it to state-of-the-art tracking methods [46–49] by evaluating them on the 3,600 hand-annotated frames of a rat surveillance video. Although the main objective was to eliminate track switches, I also evaluated the instance segmentation quality of the

[1]On a single GPU of a server equipped with two Nvidia TITAN RTX GPUs @ 24 GB memory/GPU, AMD Ryzen TR2920X CPU @ 64 GB RAM

methods (where it was applicable). For evaluations, we used two servers [2] one with two NVIDIA TITAN RTX GPUs with 24 GB memory/GPU, AMD Ryzen TR2920X@3.5GHz 24 core CPU, and 64GB RAM, and the other having two NVIDIA GeForce RTX 3090 GPUs with 24 GB memory/GPU, AMD Ryzen TR1920X@3.5GHz 24 core CPU, and 64GB RAM. Both servers had Ubuntu 18.04 operating system, codes were run in an Apptainer [50] virtual environment using Pytorch 1.4 and 1.9 for Edge Detection and Edge Completion networks respectively, with OpenCV 4.1.1. (see requirements and environment script file in the attached code).

3 New Scientific Results

THESIS I. A mapping of particle filtering onto the GPU architecture that can preserve local connections to prevent information loss

Particle filters can be considered as an extension of the Kalman filter, and therefore represent an attractive solution for Hidden Markov Model based problems in several fields, including image processing, robotics, and stock market forecasts. For computationally demanding approaches or applications required to run real time the GPU architecture allows efficient implementations. However, the resampling step of the original particle filter algorithm was considered unsuitable for parallelization without information loss, meaning a considerable limitation for speed-up. The algorithm presented in [27] exploits local connectivity of FPGA or Cellular Neural Network(CNN) chips. While other parallelized methods and distributed particle filters achieve high speeds with reduced information sharing between particles at the expense of accuracy, the algorithm presented in [27] is capable of better accuracy than the usual particle filter while drastically reducing the runtime.

In contrast to CNN and FPGA architectures GPUs, which are highly data-parallel, are widely spread devices that enable efficient computations. There have been some former implementations to GPUs, but the speed-up is highly limited at a cost of information loss in the resampling step.

[2] *nipg8* and *nipg10* of Neural Information Processing Group, Department of Artificial Intelligence, Faculty of Informatics, Eötvös Loránd University

THESIS I.1. I designed an algorithm to map the Cellular Particle Filter (CPF) to the GPU architecture, in a way to exploit the GPU memory architecture efficiently to maximize speed, and at the same time maintaining the local connection property of the original CPF topology.

I developed a method, based on the cellular particle filtering structure, to provide information sharing, for GPU architecture. Shared memory with coalesced threads and synchronization on the global memory provides a faster computation than surface or texture memory and is not limited to 2D layouts. Therefore, I developed a method to map the local connectivity of a two-dimensional CNN architecture efficiently to the one-dimensional, read/write, fast-access on-chip memory of the GPU architecture (see Figure 3 for illustration of concept).

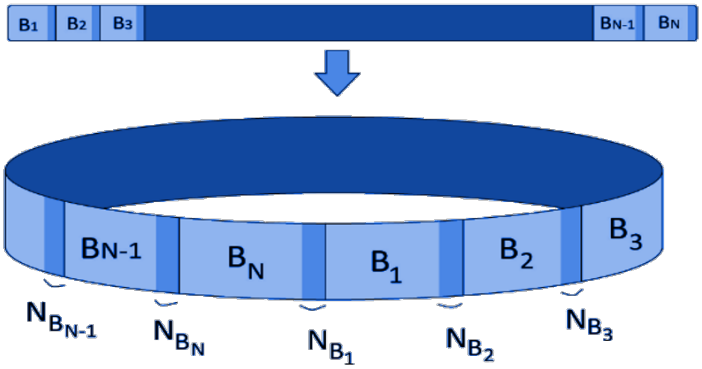

Figure 3: Restructuring linear representation of N blocks to a ring type topology. B_i stands for the i^{th} block, and N_{B_i} for the corresponding neighbourhood from the previous block, $i \in 1, \ldots, N$.

I showed by experimental evidence that the position error is similar to existing implementations on GPU, while the speed is better, although the possibility of one-to-one comparisons is limited by the variety of GPU devices used for measurements. Also, state-of-the-art works only show kernel runtimes. Those that use a device with similar performance to ours or provide no details about the used device are in the same range as our 77 ms total runtime[3], which includes I/O operations. Compared to GPU-implemented distributed particle filters, our algorithm preserves the local connectivity of the particles, therefore it achieves the accuracy of the original filter, however with a total running time of less than 12 milliseconds at 16 thousand particles per state, which corresponds to a 164x speed-up compared to CPU implementation[4]. Meanwhile, the method of mapping the 2D layout of the processors and the preservation of the

[3]Measurements were done on a NVIDIA GeForce GTX 550 Ti GPU with CUDA toolkit 4.1, available at the time of the research in 2013

[4]The following NVCC compiler options were used to drive the GPU binary code generation, as

local connections to the 1D memory architecture allows other algorithms based on two-dimensional connections to be efficiently mapped to GPU.

Related publications of the Author: [A1, A3]

THESIS II. Unsupervised segmentation of highly similar occluding rat instances built on a pipeline of deep networks exploiting edge information

In terms of overall pipeline efficiency not only the implementation of each algorithm should be considered, but also the amount of human attention and time required. Identity tracking and instance segmentation are crucial in several areas of biological research. Behavior analysis of individuals in groups of similar animals is a task that emerges frequently in agriculture, pharmaceutical studies or behavioral ecology, among others, and usually requires a decent amount of human annotation. Automated annotation of many hours of surveillance videos can facilitate a large number of biological experiments, which otherwise would not be feasible. Solutions based on machine learning generally perform well in tracking and instance segmentation; however, in the case of identical, unmarked instances (e.g., white rats or mice), even state-of-the-art approaches can frequently fail, as shown in the number of track switches listed in the second column of Table 2. The challenging task of segmenting highly similar adjacent image regions can be addressed by traditional methods [A4, A5], but deep-learning-based methods offer a more promising direction.

We focus on data where mice/rats are very similar without any markers but may have received different medical treatments, and individual behavior patterns should be analyzed. In typical setups, the camera is fixed, and the foreground segmentation is feasible, which simplifies the segmentation process. However, handling the changes in shape configurations and the heavy occlusions between the instances poses a significant challenge.

My approach is to build a pipeline inspired by *Composite AI* [33, 34], to provide reliable segmentation for identity tracking. Composite AI combines different learning architectures to achieve superior results and exploits human knowledge to improve overall performance.

The required time for tracking similar instances with human observation for each and every frame is highly time-consuming. In comparison, a partially automated solution can provide a significant speed-up. However, the time needed for the preliminary annotation for training and for the supervision in the prediction stage may still be

proposed in the CUDA SDK Guide: -arch=sm_20; -use_fast_math. We also made some measurements with -arch=sm_13. The host c code was compiled with GCC 4.5; the compiler optimization flag was -O2.

considerable for each setup. I created a pipeline with an automatic annotation method to reduce the human need in the process. My method is illustrated in Fig. 4.

Evaluations show that a tracking method built on the segmentation using the predictions of the trained models further reduces the required supervision during the prediction stage compared to state-of-the-art methods.

THESIS II.1. I developed a method to generate a synthetic dataset fully automatically without any human annotation, based on only frames with non-occluding instances, to train an edge detection network for the detection of the separating boundary between highly similar occluding rat instances with a static background.

For the 3600-frame demonstration footage of the two rats, the instance annotation for tracking is a low-skilled task for a human annotator, taking twice the time of the video. For hours of surveillance videos, this is a significant amount of time that an expert must invest before performing the tasks that require expertise (analysis and inference).

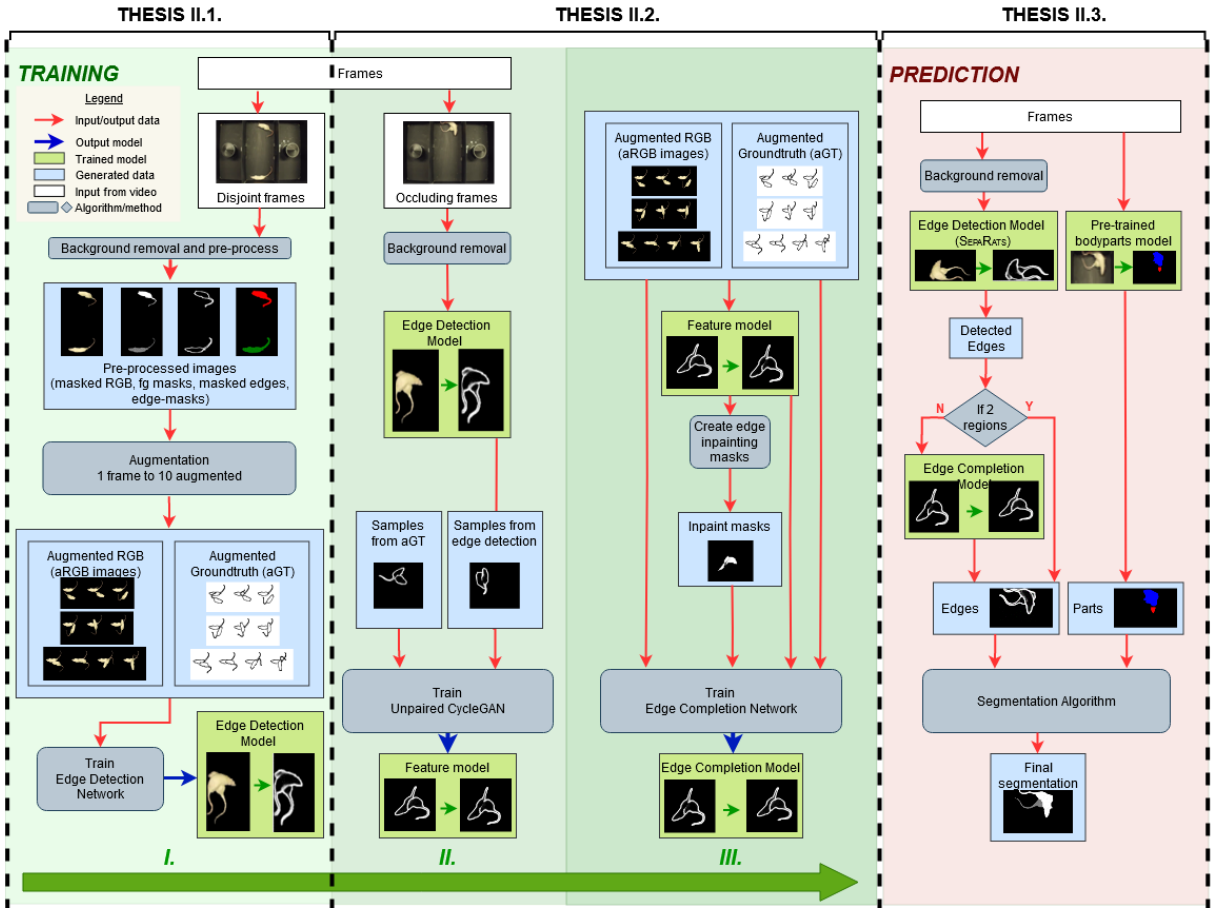

Figure 4: Illustration of the training and prediction pipelines. Three main blocks of the training pipeline: I. synthetic data generation and training Edge Detection model; II. training the Feature Model; III. extending synthetic dataset and training the Edge Completion model. Training is built upon synthetic data generation. Overlapping inputs and augmented ground truth data are constructed from pre-processed frames with non-occluding instances. The trained Edge Detection network is applied on frames with occluding instances. The unpaired CycleGAN [51] generates training data on the synthetic dataset for training the Edge Completion network. The prediction pipeline applies the Edge Detection model for the foreground of the frames. If the detected edges are not separating the instances, the foreground mask is a single connected region and the trained Edge Completion network "extends" the edges inside the foreground mask. The segmentation algorithm predicts the final segmentation for each frame. The edge regions and the body regions detected by a pre-trained model [35] are combined to provide a reliable segmentation for identity tracking of the highly similar and markerless instances including during heavy overlaps. The proposed pipeline is completely automatic, no human annotation is required. For more details, see the text.

Moreover, this speed is achieved with the trade-off of using only one in six images, meaning that the temporal resolution, thus overall tracking quality, is degraded compared to a framewise method. A trained deep network can significantly speed up

processing videos if it provides a reliable automatic instance segmentation. However, manual annotation of training data is a long, monotonous, and usually low-skilled task. Moreover, it is hardly flexible to changes in the annotation protocol and thus can be revisited when new sub-tasks or changes in recording settings occur to ensure prediction quality.

We may synthetically generate training data based on instances that can be segmented with high confidence in some of the frames automatically, i.e. with foreground segmentation. However, it is essential that the synthetic data created by the augmentation method is sufficiently similar to real frames for the deep network.

For very similar objects to create realistic occluding instances as an RGB training input, it is a challenge to accurately define the object masks in frames with not-occluding instances without losing important details, while noise from background pixels or shadows should not be added. Based on the background segmentation and that the number of instances is known and constant throughout the video my algorithm selects the frames with non-occluding foreground objects automatically. The edge images are also masked using foreground segmentation. An edge detection network [42] is trained with constructed data only. For each real RGB input of not-occluding instances, I generated 10 different overlapping positions, with slightly randomized parameters. For each position, I created both the augmented RGB (aRGB) and the augmented ground truth edge image (aGT). To overcome the noise in the aRGB images on the inner contours of the overlapping instances I applied an inpaint-based blur along the contour of the upper object. To further improve training quality I removed the background from both the aRGB and the aGT image.

With the trained model, within the frames with overlapping instances, the number of frames where the contour of the upper instance was not closed was reduced from 81.53% to 17.83%. This corresponds to a 4.57-fold improvement compared to the pre-trained state-of-the-art edge detection model (which is better than the traditional Canny [52], Sobel [53] methods).

THESIS II.2. I designed a method with deep generative networks in a self-supervised approach to improve the separating contours in an edge image of a frame with overlapping objects.

I aim to further decrease the number of frames with occluding instances in which the contour of the upper instance is not closed after applying the model of Thesis I.1. I chose a deep generative network for edge inpainting as the missing part cannot be closed by applying binary morphological operators. The required training data was generated

without human annotation to maintain the self-superwised pipeline. Note that in 3D, the unoccluded instance is above the other one. This unoccluded instance will be called the upper instance. I use a Generative Adversarial Edge Completion Network, based on [44] for the frames where the contour, detected by my Edge Detection model, of the upper instance was not closed. To train the Edge Completion network edge images with not-closed contours are required, with a mask defining the region of missing edges. I generated the edge images with not-closed contours with an unpaired CycleGAN [51], which is able to learn features of the images. Typically this network is used for learning RGB features. My idea was that features of edge images of a given type of object could also be learned by this architecture. I used the aGT edge images and the predictions of the Edge Detection model on real frames containing occluding instances. Thus, the generated not-closed contours are similar to the output of the edge detection, and the corresponding (augmented) ground-truth edge images are provided by construction. The mask of the edge region is based on the foreground mask of the instances and the generated edge image with not-closed contours.

As opposed to the inpainting technique in the original application of the [44] network, I apply no masks on the RGB input. During training the missing contour is available based on the corresponding aGT image, therefore I modified the loss function of the discriminator network. The loss of the foreground and the loss of the missing contour contribute to the overall loss equally. The Edge Completion model, from training the Edge Completion Network as described above, further reduces the frames where the upper instance has non-closed contour.

THESIS II.3. I created a segmentation algorithm that combines detected edges and detected body part regions from an existing unsupervised segmentation model to provide instance segmentation.

I used the Edge Detection and Completion models as follows. I estimated the foreground mask and generated edge images. If it was a single connected region then the algorithm of Edge Completion was invoked.

In the case of a single connected foreground region only the *body parts* method [35] was used. If the foreground mask had more than one region then we combined it with the results of the *body parts* method. An initial labeling is created for the regions created by the detected edges, after noise reduction. Edge-defined regions and regions from body part detection are assigned to each other based on the overlaps. This labeling provides the anchor regions of the watershed algorithm [54], applied within

the foreground region. The outcome is the *per-frame segmentation*. The overview of the main steps is shown in Figure 5.

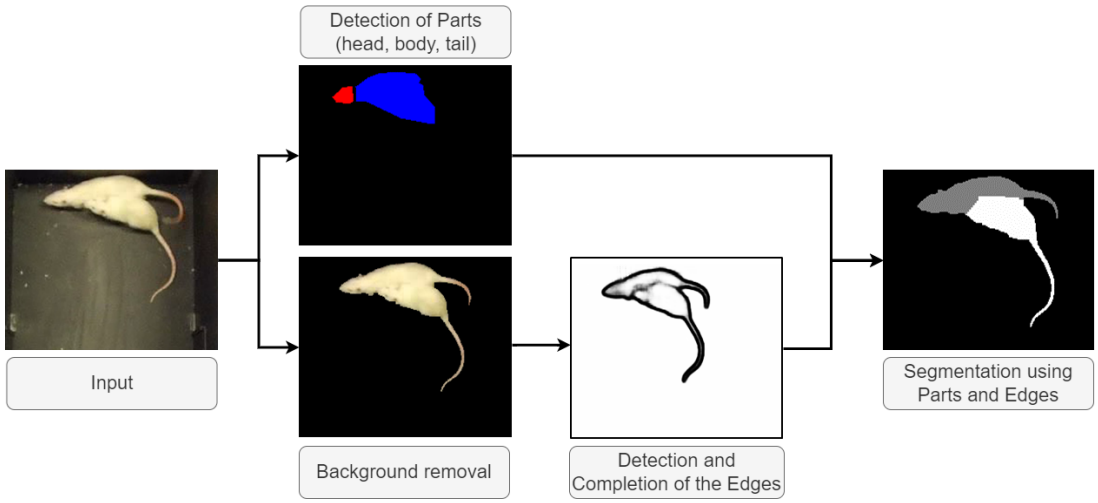

Figure 5: Sketch of the test pipeline for a single frame. Our aim is to maximize the segmentation precision within the frame to provide a strong basis for tracking. Body parts and edges are predicted and edge completion methods are invoked. A pre-processing provides the foreground masks and removes the background from the frame. A post-processing module combines the information and predicts the segmentation of the instances separately to enable identity tracking. Note the error in the last subfigure: the head of one of the animals is mislabeled. However, tracking remains error-free.

Per-frame segmentations were connected by a propagation algorithm [36] to measure the reliability of tracking based on the segmentation. I compared my results to three state-of-the-art identity tracking methods of similar kinds. No trajectory switches occurred for my method, whereas the competitive methods made several mistakes, and are outperformed in identity tracking and instance segmentation of unmarked rats in real-world laboratory video recordings. Results are presented in Table 2.

Related publication of the Author: [A2]

The codes related to the theses are available at `https://github.com/g-h-anna/phd-diss-code`.

Table 2: Comparison of segmentation-based trajectory tracking of methods which do not require prior data annotation, on our test data of 3600 frames. TS denotes the track switches metric from [36], which measures the number of ID switches during tracking. Lost ID: there are less than two different foreground labels in the frame. For ToxTrac and idtracker, per frame segmentation masks are not available. Superscript 1 marks that the result is computed by means of the GUIs of the cited method, the best we could do for comparisons. BIPED-TL is the model trained with the augmented dataset with transfer learning on the original BIPED model [42]. Gray highlight: Mean of the Intersection over Union (IoU) values for all frames. For benchmark measures, we used the same ones as in [36].

ID Tracking Results							
Approach	**Num. of TS.**	**Num. of Frames with Lost IDs**	**IoU Mean**	**IoU & F Mean**	**IoU Recall**	**F Mean**	**F Recall**
ToxTrac [48]	9	267[1]	N/A	N/A	N/A	N/A	N/A
idtracker [46]	8	1055[1]	N/A	N/A	N/A	N/A	N/A
idtracker.ai [47]	10	1485	0.5556	0.604	0.59	0.652	0.746
BIPED & Parts	4	2	0.833	0.871	0.978	0.908	0.985
SEPARATS & Parts	0	0	0.846	0.883	0.994	0.921	1.000
BIPED-TL & Parts	0	0	0.845	0.883	0.994	0.921	0.999

4 Application of the Results

The proposed deep pipeline of highly similar instance segmentation during occlusions outperformed similar approaches in segmentation and tracking quality on the annotated 3 600 images. After extending training data with a brightness-related color augmentation, it also tracked the instances without any track switches on a 224 577-frame dataset of four videos without annotated segmentations. Although this corresponds to a generalized edge detection model that required still no human annotation to deliver reliable tracking for a different setup (using another type of background or even instances that look differently), there are scenarios when a different approach can lead to a better solution. Deep models provide the highest prediction quality for data with the most similar distribution to the training data [55].Therefore, for a new data set of frames from a video clip containing consecutive scenes, which can be considered identically distributed data, we can expect the best prediction quality if the training images are most similar to the frames and, as such, have a similar distribution. We can make use of the great advantage of the pipeline that it generates training data fully automatically if provided with a set of frames of occluding and a set of frames with non-occluding instances, thus, requiring minimal human time and attention to obtain a training set adapted to the current input. With training data obtained through synthetic data generation and automatic labeling, we can train the edge detection model on highly similar images as those in the input video to get high-quality predicted segmentation. Therefore the proposed pipeline is expected to be well applicable for tasks to provide instance segmentation for id-tracking multiple, unmarked, sometimes occluding, highly similar instances with constant background and fixed distance from a fixed upper camera, if light conditions allow for the observation of visual properties (i.e. shadows) between the animals during occlusions, with a low amount of human attention and effort required.

Particle filtering is applied in multiple fields where the Kalman filter is suboptimal for state estimation due to the nonlinearity of the state dynamics and non-Gaussian noise. The proposed GPU Cellular Particle Filter retains the advantage of the original particle filtering being not limited to a specific domain. Be it image processing, autonomous driving, robotics, or any field where the particle filtering approach would lead to a solution, the proposed method can be exploited to achieve the same prediction quality with a more efficient, parallel adaption. The proposed adapted filter can be easily applied for one and two-dimensional inputs (as shown in the evaluation section for the two benchmark models) and can be modified to deal with higher dimensions.

In our proposed method, the information sharing ratio is tunable and may be modulated adaptively. Therefore, it broadens the range of options, than using a predefined information share value to find the optimal share ratio range among particles to achieve the lowest error at the highest speed. For tasks that use detections generated on the GPU device by a deep model and particle filtering is required for tracking due to the noise or other characteristics, the GPU-adapted Cellular Particle Filter can be applied with a comparable estimation quality of the CPU single-threaded PF, but spare the time-consuming data transfer between GPU and CPU. Compared to the Fermi architecture with a compute capability of 2.x available by the time of the research, the Ampere GPU architecture introduced in 2020 with a compute capability of 8.x, offers significantly higher efficiency and speed. While retaining the main concept of the algorithm, the performance of Cellular Particle Filter on GPU would be increased due to several reasons, such as the changes regarding the shared memory. The faster memory access decreases the overhead of data transfers and computations within the shared memory, and shared memory size is three times larger on an Ampere device [56] than on a Fermi [57] thus, can reduce the number of global memory accesses for synchronization of the particles. Moreover, the memory bandwidth between memory types is also significantly increased in Ampere architecture compared to Fermi architecture, not only on-device but also the PCIe-v4 buses double the speed between CPU and GPU, memory bandwidth is 1.6TB/s (for A100 40GB)–3TB/s (for H100) compared to 192GB/s of Fermi. The proposed method enables parallel execution on modern GPUs, just as it did on architectures available 10 years ago. In addition to the numerical acceleration resulting from architectural advancements the implementation of the GPU CPF algorithm on current hardware is expected to maintain a similar magnitude difference compared to the implementation of the sequential CPU PF algorithm.

In my current and future research, I aim to reduce human time in multi-animal annotation and tracking. In my approach, deep models are trained or fine-tuned on synthetically generated training samples and, based on the deep models and traditional computer vision algorithms, I propose id-tracking for multiple ruff instances in side-view recordings with several occluding positions in collaboration with the Max Planck Institute for Biological Intelligence.

Acknowledgements

First of all, I would like to express my gratitude to my supervisors. I would like to thank György Cserey that he supported me to take the first steps towards research, and I could always rely on his positive and supportive personality, which helped me face and overcome the challenges that arose. His unwavering encouragement and belief in me gave me the confidence to persevere.

I am very grateful to Kristóf Karacs for his constant guidance and support. His insights, encouragement, and our weekly meetings and consultations have been invaluable to me. He has helped me as a mentor to find my way, feel capable, and grow over the years regarding both research and education. Throughout my work, his approachable and collegial attitude has been a constant source of motivation, providing me with a sense of support that I deeply appreciate. I look forward to continuing to learn from and work with him in the future.

I would like to express my gratitude to Professor András Lőrincz, who gave me the opportunity to join the highly inspiring *Neural Information Processing Group* (NIPG) group. Without his time, advice, and professional guidance the results of Thesis group II. would not have been accomplished. He gave me considerable freedom in my research direction, while still always demanded- and guided me toward high standards. I am grateful for his thoughtful support in not only guiding me through research questions but also considering my long-term goals, and I look forward for continuing our collaboration.

I would like to thank Judit Nyékyné Gaizler, Péter Szolgay, and Kristóf Iván deans for all their support, and the opportunity to carry out my research at the University. I am very grateful to Gábor Tornai, for all professional advice during my CPF research, and to András Horváth for his help in discussing the particle filter algorithms. It has been a fantastic opportunity to work together with the members of the NIPG group, especially with László Kopácsi, Áron Fóthi, Zsombor Fülöp, Viktor Varga, Kinga Faragó, Gergő Ungvári, Ervin Téglás, Kevin Hartyáni, Milán Szász, Bálint Kovács. The last two years, with all our joint projects and discussions, were very valuable for me.

I would like to thank the Department of Artificial Intelligence of ELTE Faculty of Informatics for all support and that I could develop my research using the NIPG servers. The support of ELTE Faculty of Science is also kindly acknowledged. I am grateful to the Max Planck Institute for Biological Intelligence for giving me the opportunity to extend and continue my research in our collaboration.

Special thanks go to the administrative members of the Faculty of Information

Technology and Bionics, for all technical support and help in administrative tasks, especially to Dr. Tivadarné Vida and Mária Babiczné Rácz.

Last and most importantly, I can not express my level of gratitude to my family. The support of my husband Andris, and the joy of my children, Julcsi and Marci, was indispensable to me, as was their perseverance and patience. I would like to thank my mother and father for all the inspiration, motivation, and faith in me. The countless hours they spent looking after my children were essential for me to work on my research.

List of independent citations

Citations of article [A1] of the author*

1. Cole, C. B., Machine Learning Methods for next Generation Sequencing Data: Applications to MLL-AF4 Leukemia and Demographic Inference, *University of Oxford, PhD thesis*, 2021.

2. Spyridon Patmanidis, Alexandros Charalampidis, George P. Papavassilopoulos, Tumor Growth Modeling: State Estimation with Maximum Likelihood and Particle Filtering, *28th Mediterranean Conference on Control and Automation (MED), IEEE*, 2020, 144-149. 10.1109/MED48518.2020.9183193.

3. Havard Heitlo Holm, Martin Sætra, Peter Jan Van Leeuwen Massively parallel implicit equal-weights particle filter for ocean drift trajectory forecasting,*Journal of Computational Physics: X. 6. 100053.*, 2020, 10.1016/j.jcpx.2020.100053.

4. Dan Crisan, Joaquin Miguez, Gonzalo Ricardo Ríos Muñoz, On the performance of parallelisation schemes for particle filtering, *EURASIP Journal on Advances in Signal Processing*, 2018.05, doi:10.1186/s13634-018-0552-x

5. David Jáuregui, Patrick Horain, Real-time 3D motion capture by monocular vision and virtual rendering., *Machine Vision and Applications, 28.*, 2017., 10.1007/s00138-017-0861-3.

6. Sile Hu, Qiaosheng Zhang, Jing Wang, Zhe Chen, Real-time particle filtering and smoothing algorithms for detecting abrupt changes in neural ensemble spike activity, *Journal of Neurophysiology, 119(4)*, 2017, 10.1152/jn.00684.2017.

7. Grigorios Mingas, Algorithms and architectures for MCMC acceleration in FPGAs, *Electrical and Electronic Engineering PhD theses, Imperial College London*, 2015, DOI:10.25560/31572

8. Achutegui, Katrin et al., *A simple scheme for the parallelization of particle filters and its application to the tracking of complex stochastic systems*, 2014, DOI:10.48550/arXiv.1407.8071

*Date of list: 2023. April 3.

List of author's publications

List of journal publications

[A1] **Anna Gelencsér-Horváth** et al. "Fast, parallel implementation of particle filtering on the GPU architecture". In: *EURASIP J. Adv. Signal Process.* 2013 (2013), p. 148. DOI: 10.1186/1687-6180-2013-148. URL: https://doi.org/10.1186/1687-6180-2013-148.

[A2] **Anna Gelencsér-Horváth** et al. "Tracking Highly Similar Rat Instances under Heavy Occlusions: An Unsupervised Deep Generative Pipeline". In: *Journal of Imaging* 8.4 (2022). ISSN: 2313-433X. DOI: 10.3390/jimaging8040109. URL: https://www.mdpi.com/2313-433X/8/4/109.

List of proceedings publications

[A3] **Anna Horváth**. "Cellular Particle Filter on GPU". In: vol. 2011-2012 Academic year. 2012, pp. 145–148. URL: https://itk.ppke.hu/uploads/articles/159/file/phd_proceedings_2012.pdf.

[A4] **Anna Horváth**. "Region-merging based on contour-structure of clusters in over-segmented image". In: vol. 2012-2013 Academic year. 2013, pp. 79–81. URL: https://itk.ppke.hu/uploads/articles/159/file/Mini_simposium_2013.pdf.

[A5] **Anna Gelencsér-Horváth**. "Using contour geometry as a merging cue in over-segmented images". In: 2014, pp. 91–94. URL: https://itk.ppke.hu/uploads/articles/159/file/Mini_simposium_2014.pdf.

References

[1] P. del Moral. "Nonlinear Filtering Using Random Particles". In: *Theory of Probability & Its Applications* 40.4 (1996), pp. 690–701. DOI: 10.1137/1140078. eprint: https://doi.org/10.1137/1140078. URL: https://doi.org/10.1137/1140078.

[2] M.S. Arulampalam et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking". In: *Signal Processing, IEEE Transactions on* 50.2 (2002), pp. 174–188. ISSN: 1053-587X.

[3] Rudolph Emil Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Transactions of the ASME–Journal of Basic Engineering* 82.Series D (1960), pp. 35–45.

[4] H.F. Lopes and R.S. Tsay. "Particle filters and Bayesian inference in financial econometrics". In: *Journal of Forecasting* 30.1 (2011), pp. 168–209. ISSN: 1099-131X.

[5] S. Chib, F. Nardari, and N. Shephard. "Markov chain Monte Carlo methods for stochastic volatility models". In: *Journal of Econometrics* 108.2 (2002), pp. 281–316. ISSN: 0304-4076.

[6] Petar Djuric and Mónica Bugallo. "Particle filtering for high-dimensional systems". In: Dec. 2013, pp. 352–355. ISBN: 978-1-4673-3144-9. DOI: 10.1109/CAMSAP.2013.6714080.

[7] N. Azzabou, N. Paragios, and F. Guichard. "Image reconstruction using particle filters and multiple hypotheses testing". In: *Image Processing, IEEE Transactions on* 19.5 (2010), pp. 1181–1190. ISSN: 1057-7149.

[8] J. Czyz, B. Ristic, and B. Macq. "A particle filter for joint detection and tracking of color objects". In: *Image and Vision Computing* 25.8 (2007), pp. 1271–1281. ISSN: 0262-8856.

[9] F. Gustafsson et al. "Particle filters for positioning, navigation, and tracking". In: *Signal Processing, IEEE Transactions on* 50.2 (2002), pp. 425–437. ISSN: 1053-587X.

[10] Adrian Dalca. "Segmentation of Nerve Bundles and Ganglia in Spine MRI using Particle Filters". PhD thesis. Mar. 2012.

[11] M. de Bruijne and M. Nielsen. "Image segmentation by shape particle filtering". In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Vol. 3. 2004, 722–725 Vol.3. DOI: 10.1109/ICPR.2004.1334630.

[12] Donka Angelova and Lyudmila Mihaylova. "Contour segmentation in 2D ultrasound medical images with particle filtering". In: *Mach. Vis. Appl.* 22 (May 2011), pp. 551–561. DOI: 10.1007/s00138-010-0261-4.

[13] Hamd Abdelali et al. "Visual Vehicle Tracking via Deep Learning and Particle Filter". In: Oct. 2020, pp. 517–526. ISBN: 978-981-15-6047-7. DOI: 10.1007/978-981-15-6048-4_45.

[14] A. Doucet and A. M. Johansen. "A tutorial on particle filtering and smoothing: fifteen years later". In: *Oxford Handbook of Nonlinear Filtering* (2008), pp. 4–6.

[15] Gustaf Hendeby, Rickard Karlsson, and Fredrik Gustafsson. "Particle Filtering: The Need for Speed". In: *EURASIP J. Adv. Signal Process* 2010 (Feb. 2010). ISSN: 1110-8657. DOI: 10.1155/2010/181403. URL: https://doi.org/10.1155/2010/181403.

[16] A.S. Bashi et al. "Distributed implementations of particle filters". In: *Sixth International Conference of Information Fusion, 2003. Proceedings of the*. Vol. 2. 2003, pp. 1164–1171. DOI: 10.1109/ICIF.2003.177369.

[17] M. Bolic, P.M. Djuric, and Sangjin Hong. "Resampling algorithms and architectures for distributed particle filters". In: *Signal Processing, IEEE Transactions on* 53.7 (2005), pp. 2442–2450. ISSN: 1053-587X. DOI: 10.1109/TSP.2005.849185.

[18] C.Y. Chu et al. "Multi-prediction particle filter for efficient parallelized implementation". In: *EURASIP Journal on Advances in Signal Processing* 2011.1 (2011), pp. 1–13.

[19] Anthony Lee et al. "On the Utility of Graphics Cards to Perform Massively Parallel Simulation of Advanced Monte Carlo Methods". In: *Journal of Computational and Graphical Statistics* 19.4 (2010). PMID: 22003276, pp. 769–789. DOI: 10.1198/jcgs.2010.10039. eprint: https://doi.org/10.1198/jcgs.2010.10039. URL: https://doi.org/10.1198/jcgs.2010.10039.

[20] Raúl Cabido et al. “Multiscale and local search methods for real time region tracking with particle filters: local search driven by adaptive scale estimation on GPUs”. In: *Machine Vision and Applications* 21 (2008), pp. 43–58.

[21] Kazuhiro Otsuka and Junji Yamato. “Fast and Robust Face Tracking for Analyzing Multiparty Face-to-Face Meetings”. In: *Machine Learning for Multimodal Interaction*. Ed. by Andrei Popescu-Belis and Rainer Stiefelhagen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 14–25. ISBN: 978-3-540-85853-9.

[22] Min-An Chao et al. “Efficient parallelized particle filter design on CUDA”. In: *2010 IEEE Workshop On Signal Processing Systems* (2010), pp. 299–304.

[23] Mehdi Chitchian et al. “Distributed Computation Particle Filters on GPU Architectures for Real-Time Control Applications”. In: *IEEE Transactions on Control Systems Technology* 21.6 (2013), pp. 2224–2238. DOI: 10.1109/TCST.2012.2234749.

[24] Olivier Brun, Vincent Teulière, and Jean-Marie Garcia. “Parallel Particle Filtering”. In: *J. Parallel Distributed Comput.* 62 (2002), pp. 1186–1202.

[25] Vasileios Belagiannis et al. “Segmentation Based Particle Filtering for Real-Time 2D Object Tracking”. In: vol. 7575. Oct. 2012, pp. 842–855. ISBN: 978-3-642-33764-2. DOI: 10.1007/978-3-642-33765-9_60.

[26] J.V. Candy. “Bootstrap Particle Filtering”. In: *Signal Processing Magazine, IEEE* 24.4 (July 2007), pp. 73–85. ISSN: 1053-5888. DOI: 10.1109/MSP.2007.4286566.

[27] A. Horvath and M. Rasonyi. “Topographic implementation of particle filters on cellular processor arrays”. In: *Elsevier Signal Processing* (2013). DOI: 10.1016/j.sigpro.2012.11.025.

[28] L. Chua and L. Yang. “Cellular Neural Networks: Theory”. In: *IEEE Trans. on Circuits and Systems* 35(10) (1988), pp. 1257–1272.

[29] Dorin Comaniciu and Peter Meer. “Meer, P.: Mean shift: A Robust Approach Toward Feature Space Analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5), 603-619”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24 (June 2002), pp. 603–619. DOI: 10.1109/34.1000236.

[30] Li Liu et al. “Deep Learning for Generic Object Detection: A Survey”. In: *International Journal of Computer Vision* 128.2 (Feb. 2020), pp. 261–318. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01247-4. URL: https://doi.org/10.1007/s11263-019-01247-4.

[31] Tobias Glasmachers. “Limits of End-to-End Learning”. In: *CoRR* abs/1704.08305 (2017). arXiv: 1704.08305. URL: http://arxiv.org/abs/1704.08305.

[32] Aston Zhang et al. “Dive into deep learning”. In: *arXiv preprint arXiv:2106.11342* (2021).

[33] Gartner. *The 4 Trends That Prevail on the Gartner Hype Cycle for AI*. https://www.gartner.com/en/articles/the-4-trends-that-prevail-on-the-gartner-hype-cycle-for-ai-2021. [Online; accessed 7-Dec-2021]. 2021.

[34] Debmalya Biswas. *Compositional AI: Fusion of AI/ML Services*. Mar. 2021. URL: https://www.researchgate.net/publication/351037326_Compositional_AI_Fusion_of_AIML_Services.

[35] László Kopácsi, Áron Fóthi, and András Lőrincz. "A Self-Supervised Method for Body Part Segmentation and Keypoint Detection of Rat Images". In: *Annales Univ. Sci. Budapest., Sect. Comp*. Vol. 53. 2021.

[36] László Kopácsi et al. "RATS: Robust Automated Tracking and Segmentation of Similar Instances". In: *Artificial Neural Networks and Machine Learning – ICANN 2021*. Springer, 2021, pp. 507–518. ISBN: 978-3-030-86365-4. DOI: 10.1007/978-3-030-86365-4_41.

[37] Bradley P. Carlin, Nicholas G. Polson, and David S. Stoffer. "A Monte Carlo Approach to Nonnormal and Nonlinear State-Space Modeling". In: *Journal of the American Statistical Association* 87.418 (1992), pp. 493–500.

[38] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". In: *IEEE Proceedings F, Radar and Signal Processing* 140.2 (1993), pp. 107–113. DOI: 10.1049/ip-f-2.1993.0015.

[39] G. Kitagawa. "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models". In: *Journal of computational and graphical statistics* 5.1 (1996), pp. 1–25.

[40] D. Salmond Gordon N. and A.F.M. Smith. "Novel approach to nonlinear/nonGaussian Bayesian state estimation". In: *IEE Proceedings* F-140 (1993), pp. 107–113.

[41] O. Morris, M. Lee, and A. Constantinides. "A unified method for segmentation and edge detection using graph theory". In: *ICASSP '86. IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 11. 1986, pp. 2051–2054. DOI: 10.1109/ICASSP.1986.1168866.

[42] Xavier Soria Poma, Edgar Riba, and Angel Sappa. "Dense Extreme Inception Network: Towards a Robust CNN Model for Edge Detection". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Mar. 2020.

[43] Xavier Soria Poma et al. "Dense Extreme Inception Network for Edge Detection". In: *CoRR* abs/2112.02250 (2021). arXiv: 2112.02250. URL: https://arxiv.org/abs/2112.02250.

[44] Kamyar Nazeri et al. "EdgeConnect: Structure Guided Image Inpainting using Edge Prediction". In: *The IEEE International Conference on Computer Vision (ICCV) Workshops*. Oct. 2019.

[45] Saining Xie and Zhuowen Tu. "Holistically-Nested Edge Detection". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1395–1403. DOI: 10.1109/ICCV.2015.164.

[46] Alfonso Pérez-Escudero et al. "IdTracker: Tracking individuals in a group by automatic identification of unmarked animals". In: *Nature methods* 11 (June 2014). DOI: 10.1038/nmeth.2994.

[47] Francisco Romero-Ferrero et al. "Idtracker. ai: tracking all individuals in small or large collectives of unmarked animals". In: *Nature methods* 16.2 (2019), pp. 179–182.

[48] Alvaro Rodriguez et al. "ToxId: an efficient algorithm to solve occlusions when tracking multiple animals". In: *Scientific reports* 7.1 (2017), pp. 1–8.

[49] Alvaro Rodriguez et al. "ToxTrac: a fast and robust software for tracking organisms". In: *Methods in Ecology and Evolution* 9.3 (2018), pp. 460–464.

[50] *Apptainer: Application containers*. [Last accessed 16-Jan-2023]. URL: https://apptainer.org/.

[51] Jun-Yan Zhu et al. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networkss". In: *Computer Vision (ICCV), 2017 IEEE International Conference*. 2017.

[52] J. Canny. "A Computational Approach to Edge Detection". In: *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*. Ed. by M. A. Fischler and O. Firschein. Los Altos, CA.: Kaufmann, 1987, pp. 184–203.

[53] Nick Kanopoulos, Nagesh Vasanthavada, and Robert L Baker. "Design of an image edge detection filter using the Sobel operator". In: *IEEE Journal of solid-state circuits* 23.2 (1988), pp. 358–367.

[54] Anton S. Kornilov and Ilia V. Safonov. "An Overview of Watershed Algorithm Implementations in Open Source Libraries". In: *Journal of Imaging* 4.10 (2018). ISSN: 2313-433X. DOI: 10.3390/jimaging4100123. URL: https://www.mdpi.com/2313-433X/4/10/123.

[55] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[56] NVIDIA Ampere architecture paper. "Accessed 2023-02-08". URL: https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf.

[57] *NVIDIA Fermi architecture paper*. "Accessed 2023-02-08". URL: https://www.nvidia.com/content/PDF/fermi%5C_white%5C_papers/NVIDIA%5C_Fermi%5C_Compute%5C_Architecture%5C_Whitepaper.pdf.