Fast Reactive Grasping with In-Finger Vision and In-Hand FPGA-accelerated CNNs

Felix Hundhausen, Raphael Grimm, Leon Stieber and Tamim Asfour

***Abstract*— We present a soft humanoid hand with in-finger integrated cameras and in-hand real-time image processing system for fast reactive grasping. Specifically, we describe an FPGA-based, in-hand integrated, embedded system for processing visual data captured by the five in-finger cameras while avoiding high bandwidth raw data streaming via the robots real-time data bus. The hardware acceleration allows fast detection and localization of objects based on finger-camera images and provides input for a grasping controller. To this end, we implement a resource-aware encoder-decoder Convolutional Neural Network (CNN) for pixel-wise object segmentation and run inference on the in-hand embedded system at 3.58 GOPS. We evaluate the system, consisting of the soft hand with in-finger vision and the in-hand FPGA-accelerated CNN in several experiments on the humanoid robot ARMAR-6. Specifically, we evaluate the overall system response time, the ability to perform precision grasps and test reactivity and reliability that are required for handover actions. We obtain an overall system response time of 154 ms for catching a falling object and obtain a success rate of 90 % reliability for the power drill handover tasks. Further, we successfully demonstrate ability of dexterous grasping and manipulation of a pencil from a cup.**

I. Introduction

Visual perception is an important sensing modality for autonomous systems ranging from self driving cars to assistance household robots. In robotic grasping, fast and reliable object perception is crucial for successful grasp execution. The problem of vision based grasping includes the three key tasks of object localization, object pose estimation, grasp planning and execution [1]. In case of a closed-loop system, visually detected features should be continuously estimated and updated during the robot and/or object motion [2], thus the latency of feature tracking constrains the maximum control frequency and thus the overall response time of the system. Aiming at robot systems that can compete with human performance regarding manipulation tasks, high demands and constraints on the real-time processing hardware and applied methods are the consequence. In particular, image processing tasks are challenging and demanding because of the high workload of the methods used to process large amounts of raw visual sensor data and extract relevant scene information including object positions.

To address vision-based grasping, numerous approaches and systems have been proposed in the literature. Eye-in-hand systems have been used for image-based visual servoing [3] to provide a direct view on the object of interest in the scene. The miniaturization of cameras allows integration of one [4] or multiple cameras [5] into robotic hands. Moreover, in recent years also the design and control of grippers and robotic hands with finger-integrated cameras is an active research area [6], [7], [8], [9].

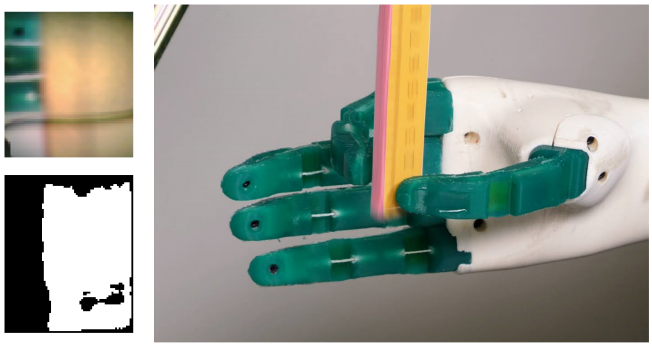

Fig. 1. Catching a falling ruler based on in-finger vision and in-hand hardware-accelerated CNN inference for object segmentation. An exemplary segmentation result (=CNN output + threshold) and RGB image obtained from an in-finger camera (thumb) are shown on the left.

However, real-time processing of data of multiple high resolution visual sensors in robotics, or more specifically a humanoid hand equipped with several high resolution cameras, such as our in-finger vision soft humanoid hand [9], is challenging. The reasons for this are the limited communication bandwidth as well as constrained computing and energy resources. The limitation of bandwidth is mainly due to the mechanical complexity and kinematic structure that e. g. makes it difficult to use multiple busses across multiple robot and hand joints due to intricate cable integration. Further, the computer architecture usually used in today's robots provides a certain number of processing units such as CPUs and GPUs for perception, planning, interaction and control tasks. While the use of multiple GPUs might be sufficient to process visual data generated by multiple high resolution cameras, this would lead to high energy consumption that is still difficult to ensure given the limited energy resources of a mobile robot. A promising solution for such problems in robotics is the use of decentralized data processing in dedicated processing hardware such as FPGAs, where sensor data can be pre-processed locally in an energy efficient and task specialized hardware while guaranteeing real-time execution.

In this work, we present a soft humanoid hand with in-finger integrated cameras and an FPGA-based, in-hand

This work has been supported by the German Federal Ministry of Education and Research (BMBF) under the project INOPRO (16SV7665), the KIT Centers pilot project SoftNeuro and by the Competence Center Karlsruhe for AI Systems Engineering (CC-KING) sponsored by the Ministry of Economic Affairs, Labour and Tourism Baden-Württemberg.

The authors are with the Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Karlsruhe, Germany. {felix.hundhausen, asfour}@kit.edu

integrated embedded system for real-time image processing needed for fast reactive grasping. To this end, we develop a resource-aware encoder-decoder Convolutional Neural Network (CNN) for visual object detection and localization that is running on the hand internal FPGA to provide segmentation masks needed for vision-based grasping tasks running on the main robot control PC. We evaluate the hardware-accelerated data processing algorithm as well as the system as a whole by a set of experiments on the humanoid robot ARMAR-6 [10] in the context of the tasks of fast grasping, human-robot handover and dexterous grasping. An example task of catching a falling ruler is shown in Fig. 1.

The paper is structured as follows: In section II we give an overview of related work and in section III we describe the design of the system including the soft humanoid hand developed and used in this work, the network architecture and its implementation. Section IV describes the performance evaluation of visual data processing and the conducted experiments on the humanoid robot ARMAR-6.

II. Related Work

To achieve the abilities of the human hand in robotic grasping and manipulation, it is important to equip artificial hands with sensors enabling environment perception. But replicating the entire multimodal somatosensory sensor system of the human hand in robotic hands is still out of reach. A large body of work address the problem of contact based tactile sensing and proprioception using sensors with different measurement principles such as capacitive [11], pressure [12], strain gauge [13], optical based [14] and acoustic [15] based sensing. In addition, *contact-less* methods based on proximity sensors (e. g. [16]) or, as in our work presented in this paper, vision using in-finger integrated visual sensors, are considered promising and hence investigated by a number of researchers and thus are discussed in the following.

The authors in [17] use finger-vision for a two finger gripper and obtain tactile as well as visual information of the scene. In [7], cameras integrated into the fingers of a parallel gripper are used to acquire proximity and contact information as well as visual feedback. In [8], additionally to finger and hand palm integrated cameras, external high speed cameras are used. The system is evaluated in the context of a ball catching task, where image processing is realized on a dedicated vision PC. In [18], a dual mode, vision based tactile sensor is presented, which allows identifying objects based on their surface appearance. In our previous work, we have presented the KIT Finger Vision Soft Hand [9] with an integrated miniature camera in each finger and a hybrid embedded processing system consisting out of an FPGA and a microcontroller. We showed that pixel-wise semantic object segmentation based on image data captured by the cameras during the different phases of the grasping process can be performed with a mean classification accuracy of more than 90% for an object set consisting of five objects.

As stated in [19], eye-in-hand setups allow, compared to static camera setups, a more precise control without a need for camera calibration. Examples of robotic grasping approaches that use in hand cameras can be found e. g. in [19], [20], [21], and [22]. These approaches use neural networks to either detect object position from visual input data ([19], [20]) or directly learn features from visual data ([21] and [22]). In [22], the authors report a runtime of $19\,\mathrm{ms}$ of the grasp detection pipeline of the reactive grasping system implemented using a GPU-equipped desktop computer.

When high execution speed is important for successful task execution as in the task of catching a ball in mid-flight, careful design of the vision and processing architecture is required. As presented in [23], [24] and [8], these task can be achieved using dedicated computers for purpose of image processing and motion control.

The use of FPGAs in robotics comes with the advantage of a possibly better performance and higher energy efficiency, that can exceed CPU and GPU based systems by a factor of 10 [25]. This makes FPGAs suitable for space constrained systems where power consumption and thus heat generation is often a critical design aspect. Thereby, FPGAs provide promising possibilities for various robotic tasks related to sensing, perception, localization, planning and control [25]. Recently, several robot hands that use FPGA-based processing architectures to pre-process sensor-data and provide high-bandwidth data output have been presented, e. g. in [26], [27], and [5].

This work focuses on CNNs for image processing. CNNs are widely used and have been proven to be very well suited for a variety of image processing tasks including image classification and segmentation. Further, CNNs can be easily adapted to new object classes or tasks and can scale in complexity concerning number of recognized features. The inference of CNNs under resource- and real-time-constraints, especially by using accelerators like FPGAs, is an active field of research. A survey of FPGA-based neural network (NN) acceleration is given in [28]. Here, the most important parameter to describe a NN accelerator design is the throughput IPS, that describes the number of network-inferences per second given by

$$IPS = \frac{OPS_{peak} \times \eta}{W},$$

where OPS_{peak} is maximum number of operations that can be processed per second, η is the utilization ratio of the processing units and W is workload for each inference, measured by the number of operations in the network. State-of-the-art hardware accelerated CNNs reach up to 3000 GOPS at 16 bit accuracy [29] and an energy efficiency of more than 100 GOP/J.

Fixed-point data representations for quantization of network weights and activations can significantly reduce processing and memory resource demands while maintaining the accuracy at almost similar level, see [30], [31] and [32]. In Xilinx logic implementation, 8 bit fixed-point representations compared to 32 bit floating-point multiplication reduces the need for number of Look-Up Tables (LUTs) by factor 9.44 and 10.73 for Flip-Flops (FF) [28], while reducing memory

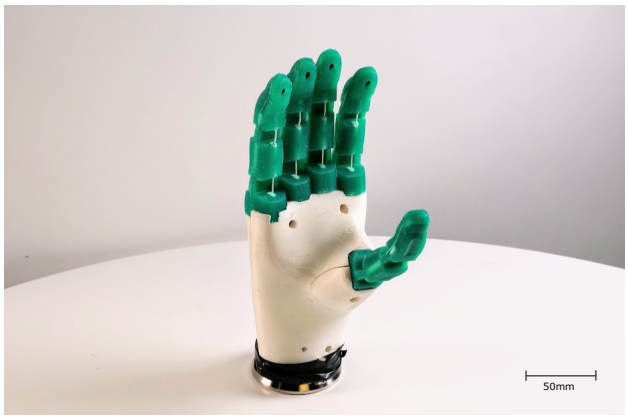

Fig. 2. The KIT Finger-Vision Soft Hand (2nd revision). The hand includes a miniature camera in each fingertip and an in-hand hybrid data processing system.

demand by a factor of four.

III. System Design

The system presented in this paper comprises several aspects: As robotic hardware platform we use the Finger-Vision Soft Hand [9] with minor changes. This work proposes a CNN design as well as a task specific hardware accelerator implemented on the hand internal FPGA that allows perception in real-time. The presented design is used with the humanoid robot ARMAR-6 [10] to control the hand and conduct several experiments for evaluation.

A. In-Finger-Vision Soft Hand

Soft robotic hands have the advantage of being able to safely handle objects by exploiting the natural dynamics of interaction with the objects. However, a precise control of such hands is a difficult problem due to the limited and often unavailable information about the state of the hand that can only be achieved by suitable sensorization of the hand.

The KIT Finger-Vision Soft Hand as shown in Fig. 2 is equipped with miniature cameras in each finger tip. As a humanoid hand it includes five fingers that are actuated by 3 DC motor gear units. Thumb and index fingers are actuated by a motor each and middle, ring and little finger are coupled by an under-actuation mechanism. The fingers are manufactured using soft material (RTV silicone, Shore hardness A45, green colored) and rigid bone segments. Each finger includes a miniature low cost camera with a maximum resolution of 2 megapixels (OmniVision OV2640). In our setting, the output of the camera is configured to an output image size of 88×72 pixels in RGB888 format. All 5 cameras are synchronized and new frames are received at equal time intervals. The frame rate of each camera is $10.4\,\mathrm{Hz}$ resulting in a total image rate of 52 images per second when all five cameras are considered. This gives a maximum timing budget for image processing of $19.2\,\mathrm{ms}$ that is available as CNN inference time including IO-overhead. Parallelization of frame processing is not intended to obtain a minimum processing delay.

The hand internal hybrid data processing system consists of a processor and an FPGA. The processor is an Arm Cortex M7 based microcontroller (ST Microelectronics, STM32H753VIH6, 11.07 €) running a clock frequency of $400\,\mathrm{MHz}$. The used FPGA from the Artix 7 series (Xilinx, XC7A75T-1CSG324C, 84.43 €) provides 75k logic cells and 3780 kBit of block RAM. Compared to available FPGAs, the XC7A75T is at the lower-end of the performance spectrum. All five camera data interfaces are directly connected to IO-pins of the FPGA, which shares a parallel interface to the processor. Further, a $100\,\mathrm{Mbit}$ real-time EtherCAT connected to the processor allows communication with the robots main control PC at a control frequency of $1000\,\mathrm{Hz}$. For capturing training data, the hand is operated manually and obtained data is forwarded to a PC for training. During the experiments conducted for evaluation, the hand is mounted at the right arm of the humanoid robot ARMAR-6.

B. Convolutional Neural Network Design

For the binary pixel-wise segmentation of objects in the in-finger camera images, a convolutional neural network in encoder-decoder architecture as shown in Fig. 3 is used. The network consists of five convolutional layers with zero padding to preserve the original input size. Two consecutive 2×2 max pooling layers perform the down-sampling in the encoder-path, while two 2×2 up-sampling layers with nearest-neighbour interpolation are utilized in the decoder-path. The final binary pixel-wise segmentation mask is generated by applying a threshold to the output. Training the network was performed class-aware, resulting in separate weight configurations for each object class. We captured between 230 and 310 images per object and split these datasets in a ratio of 9/10 and 1/10 for training and testing. Recording of training and test data set as well as the experiments was conducted in a typical lab-environment, with a white photo-background only on one side behind the robot, see Fig. 6 c). For data augmentation of training images we use horizontal and vertical flipping but no further augmentation was used to keep the data close to the real use case. The labels were generated manually by creating a binary ground truth mask for every image. To account for a high imbalance in the amount of background and object pixels, we trained the network with the Dice loss function. The resulting mean accuracy on the test dataset for the included objects show variations due to the different appearance of the objects. We obtain the following accuracies of $83.7\,\%$ for the ruler, $94.3\,\%$ for the power drill and $96.0\,\%$ for the pencil.

Initially, the weights and activations of the segmentation network are represented as 32 bit floating-point numbers. However, using quantization, the representation is changed to 8 bit fixed-point numbers leading to significant reduction of required processing and memory resource demand of the network while maintaining the accuracy at a similar level. To achieve this, we apply the method described in [32]. We iteratively identify a proper fixed-point format for weights, activations and biases in each layer based on the respective minimum and maximum values and induced accuracy drop on the training dataset. To convert between different fixed-point formats in consecutive layers, we utilize

bit-shifts, avoiding time-consuming de- and re-quantization of activations during inference. The resulting accuracy drop caused by quantization noise for the used objects is $0.2\,\%$ for the pencil and $0.8\,\%$ for the powerdrill. For the segmentation of the ruler, the accuracy was increased by $1.1\,\%$, which could come from additional regularization caused by the quantization.

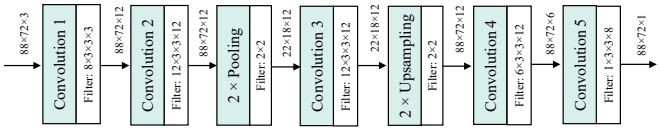

Fig. 3. Encoder-decoder network architecture

C. CNN Hardware implementation

The accelerator design implemented in FPGA-hardware as depicted in Fig. 4 is specifically optimized for inference of a convolutional neural network in encoder-decoder architecture. Our implementation includes a program control component that enables and parametrizes the processing units and manages memory and interface accesses. Using a communication interface to the microcontroller object specific weight parameters, stored in the microcontrollers flash memory are obtained. The detected object type and thereby specific weight set are selected by the main robot control unit via the bus interface of the hand.

1) Processing units: The convolution filter kernel and bias values received from the microcontroller are stored in a double buffered shift register. This allows to receive weights for the next layer without any delay resulting from IO-overhead in best case. The total memory size of all kernel, bias and output shifts are 112 byte for a 3×3 kernel for one output feature and 12 input features.

The convolution unit is implemented using the FPGA integrated DSP units to reduce the demand for general purpose FPGA fabric. The convolution unit is designed to process one output pixel for every clock cycle, implying for each processing step data from every input feature maps is required. Therefore, we use the common implementation scheme of buffering 3 lines of the input feature maps and shift the data through the shift register as shown in Fig. 5. This avoids multiple reading of input data and reduces memory interface bandwidth. In total, our convolution unit includes 108 DSP units that allow to process up to 12 input feature maps using 3×3 kernels. The CNN was designed to meet these constraints. The implemented convolution unit design can be used with a clock rate of $50\,\mathrm{MHz}$ that results in OPS_{peak} = 5.4 GOPS at continuous and complete resource utilization.

The pooling and up-sampling units are implemented using 2×2 filters. We used max pooling for the pooling layers and nearest-neighbour interpolation for the up-sampling layers.

2) Memory management: For storing camera image data and processed feature output maps, our design makes use of the FPGA internal block ram, where in total 1247 kByte are required. The memory is partitioned in 7 separate banks, one

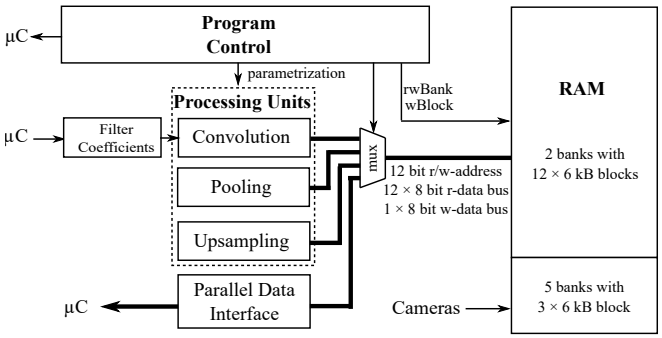

Fig. 4. System diagram of the CNN implementation in the FPGA.

for each camera input stream consisting of 3×6336 byte that can contain the R, G and B channels of the 88×72 pixel input image. Two larger banks consisting of 12×6336 byte blocks are used as working memory for output feature maps, where one bank functions as read-memory with 12 parallel read busses and the other as write-memory interfaced by one parallel bus. The read and write block are swapped after every completely processed layer.

D. CNN Processor Implementation

As a reference for the FPGA design, we implement the CNN inference on the in-hand available Arm Cortex M7 based high-performance microcontroller. In our previous work, we used this controller for CNN based classification with an acceptable processing time, while the CNN used for segmentation could not satisfy real-time requirements [33]. For inference on the processor, the network is implemented in C-code and compiled with the optimization level -Ofast (optimize for speed). The implementation for the processor uses the same activation-shifts and 8 bit fixed-point weights and biases like the FPGA hardware implementation.

IV. Experimental Evaluation

To evaluate and test the applicability for real world tasks of the proposed system and processing architecture, we evaluate the performance of the processing system and conduct a set of experiments. This includes fast reactive grasping of a ruler, handover of a power drill and dexterous grasping of a pencil with the hand attached to the right arm of the humanoid robot ARMAR-6.

A. Image Processing Performance

We evaluate the performance of the in hardware implemented image processing system, i. e. the inference of the network architecture described in Section III-B. The total

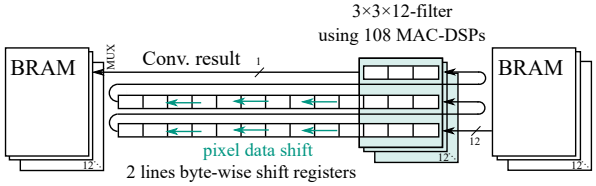

Fig. 5. Convolution Unit Design

workload of the network is 16.77 MOPS regarding only the multiplication operations required for convolution. As a total inference time of the accelerator implemented in FPGA-hardware, we obtain an inference run-time of $5.46\,\mathrm{ms}$. The convolution layers proportion of the run-time is at $77.9\,\%$, IO-overhead for obtaining weight data from the microcontroller accounts for $16\,\%$ of the run-time and could be further reduced by an optimized implementation if needed. The resource utilization of the DSP units during convolution correlates with the number of input feature maps, in case of less than 12 input features, the memory interface limits the design performance. At maximum usage of the DSP units, 5.132 GOPS are achieved in the 2^{nd} and 4^{th} convolution layers. The mean number of operations during complete inference of the network is obtained as 3.58 GOPS. The power consumption P of the implemented design amounts $0.244\,\mathrm{W}$ as total estimated chip power. The static power is $0.101\,\mathrm{W}$ $(41\,\%)$ and the dynamic power is $0.143\,\mathrm{W}$. RAM and DSPs account for $29\,\%$ respectively $31\,\%$ of the dynamic power. The on-chip energy efficiency of the hardware accelerator ($\frac{OPS_{FPGA}}{P}$) is reached as $14.7\,\mathrm{GOP/J}$. The total required energy per inference E_{total} is obtained as $1.33\,\mathrm{mJ}$. The reference implementation of the microcontroller reaches an inference run-time of $1358\,\mathrm{ms}$ with a proportion of $99.7\,\%$ convolution layer. The total speedup factor during convolution of the hardware implementation compared to processor-based implementation is 318 and 247 during the whole network inference. The highest speedups (> 400) can be achieved for the maximum size convolution layer that have highest FPGA resource utilization. For pooling layer, the speedup is only insignificant, however it does not make up a large portion of microprocessor run-time. Results in detail are summarized in Table I. When comparing speedup to cost ratio, the almost 8 times more expensive FPGA part has still a $32\times$ higher performance to cost ratio.

B. Robot experiments

For the robot experiments, the Finger-Vision Soft Hand is attached to the right arm of ARMAR-6 and three experiments are performed: fast reactive grasping of a falling ruler, handover of a power drill from a human to the robot and dexterous grasping of a pencil. The robot is controlled by the robot software framework ArmarX [34].

1) Catching a falling ruler: To estimate the total response time of the system, we conducted an experiment, where a ruler is dropped between the thumb and index finger of the hand and the task is to detect and catch the ruler during falling under gravitational acceleration. The fallen distance d is used to calculate system response time $t_{response}$. Assuming gravitational acceleration $g = 9.81\,\mathrm{m/s^2}$ for the falling ruler and neglecting air resistance, the response time is given by $t_{response} = \sqrt{2d/g}$.

The overall system response time comprises camera exposure time, image processing time, bus transmission delays and time required for the execution of finger motion. As described in Section III-B, object specific CNN weight parameters are used for visual segmentation of the ruler.

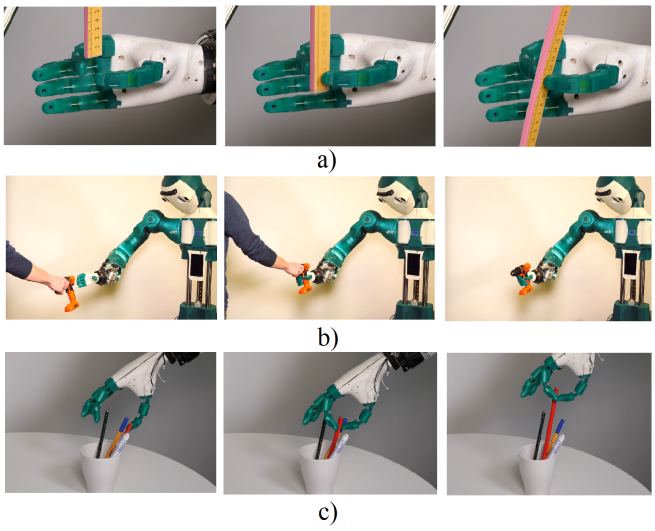

Fig. 6. Conducted experiments using only in-finger visual data: a) Test of system response time in catching a falling ruler b) reliability in executing a human-robot handover task c) ability of dexterous grasping and manipulation. The experiment setup includes a white paper-sheet behind the robot. Apart from that, the environment is a typical lab-environment.

For detection, positive classified pixels in all finger cameras are counted and an experimentally determined detection threshold is applied to detect the ruler in-between the fingers. The ruler is held and dropped from above a few millimeters between thumb and index finger so that the ruler is not visible in the finger camera-images before the drop. Photos of the experiment are shown in Fig. 6 a).

13 consecutive trials were conducted to achieve 10 successful catches, in the remaining 3 cases the ruler was not caught by the robot. Failed attempts are caused by unsuitable finger position or data transmission errors. To compare this with human performance, we conducted the experiment in a similar setup with 4 humans as test persons that were granted 3 test trails before collecting their response time data for 10 trials.

The results of the experiment are shown in Fig. 7. The robot's median response time amounts $154\,\mathrm{ms}$ ($\pm$ 30.1 ms). At our test setup, the human test persons reach a response time of $168\,\mathrm{ms}$ to $180\,\mathrm{ms}$. In literature, measurement of reaction time for grasping a falling ruler can be found for a slightly varied setup. In [35] a repeated assessment of reaction times is performed where the obtained times range from $252\,\mathrm{ms}$ to $265\,\mathrm{ms}$ for persons at age 21.8 $\pm$2.6 years (n=43). As it can be seen, the robot outperforms reaction time of humans in our test by $14\,\mathrm{ms}$ to $26\,\mathrm{ms}$ (median). However, standard deviation for the robot is larger than for the test persons, probably resulting from the not event triggered perception and additional sensor noise.

2) Human-Robot Handover: We test the reliability of the overall system during a human-robot handover task, which is important for human-robot collaboration. The robot should detect and successfully grasp the object that is passed to it by the human as shown in Fig. 6 b). Dropping the object or not being able to detect it are considered failure of the

Layer	Workload W (Multiplications)	Runtime Processor	$OPS_{Processor}$	Runtime FPGA (including IO-overhead)	FPGA resource utilization	OPS_{FPGA}	FPGA speedup
Convolution 1	2.05 MOP	282 ms	7.279 MOPS	1.600 ms (+0.076 ms)	25 %	1.283 GOPS	× 176
Convolution 2	8.21 MOP	660 ms	12.44 MOPS	1.600 ms (+0 ms)	100 %	5.132 GOPS	× 413
Pooling 1	Pool to 36 × 44	2.25 ms		0.127 ms (+0 ms)			× 18
Pooling 2	Pool to 18 × 22	0.58 ms		0,032 ms (+0 ms)			× 18
Convolution 3	2.05 MOP	41 ms	12.51 MOPS	0.120 ms (+0.810 ms)	100 %	4.276 GOPS	× 342
Upsampling 1	Up to 36 × 44	0.14 ms		0.032 ms (+0 ms)			× 4
Upsampling 2	Up to 72 × 88	0.55 ms		0.127 ms (+0 ms)			× 4
Convolution 4	4.11 MOP	338 ms	12.15 MOPS	0.800 ms (+0 ms)	100 %	5.132 GOPS	× 423
Convolution 5	0.34 MOP	33 ms	10.37 MOPS	0.133 ms (+0 ms)	50 %	2.572 GOPS	× 248
Total	16.77 MOP	1358 ms 1354 ms (only Conv.)	11.24 MOPS (mean)	5.46 ms (incl. IO-Ov.) 4.25 ms (only Conv.)		3.580 GOPS	× 247 × 318 (only Conv.)

TABLE I

PERFORMANCE RESULTS OBTAINED FOR INFERENCE OF THE CNN WHEN EXECUTED BY THE FPGA COMPARED TO MICROPROCESSOR.

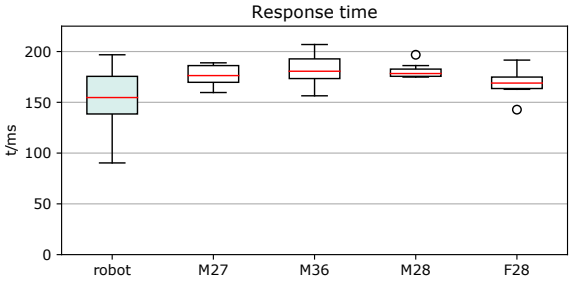

Fig. 7. System response time calculated based on the fall distance and gravitational acceleration of a ruler that is caught by the hand as soon as detected. The resulting time for the robot and four test persons performing the same task are shown.

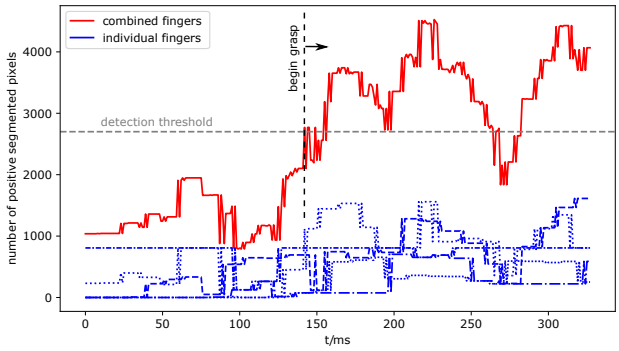

Fig. 8. Number of positive segmented pixels during grasping of a pencil. Note: Not all images were transferred correctly, however the pencil was successfully detected and grasped.

system. We use the power drill from the YCB object set and conduct a series of 10 repetitive human-robot handover tasks. During the experiments, our system reaches a success-rate of 90%, one out of 10 trials failed due to the object not being detected. All remaining handovers were successfully executed, i. e. without dropping the power drill.

3) Dexterous Grasping: To test the ability of dexterous grasping and manipulation of small objects, we implement a controller to execute a precision grasp using the thumb and index finger, while the remaining fingers are kept in open position. The experimental setup includes a cup with four different types of pencils as shown in Fig. 6 c). The position of the cup is known to the robot while the position of the pencils is not known in advance. However, all pencils are reachable with a linear motion of the tool center point of the hand of 20 cm across the cup position. Before starting the motion, an index-thumb pre-shape of the hand is selected. As soon as the target pencil is visually detected, a suitable grasp is executed and the pencil is lifted out of the cup. The detection is based on number of positive segmented pixels in the in-finger camera images. If a threshold determined in advance is exceeded, movement of the arm is halted and the pencil is grasped and lifted. In Fig. 8, the number of positive segmented pixels as well as the detection threshold is plotted. Despite significant sensor noise, such as blurred images, the pencil can be grasped successfully when parameters like finger pose and arm movement speed are adjusted accordingly. As we describe in our previous work [9], when the fingers approach and touch the object, the segmentation accuracy decreases and prediction become less reliable.

V. LIMITATIONS AND FUTURE WORK

In this paper, we only use information obtained in camera coordinates on which we base the decision to start the execution of a grasp. In future work, we aim at using 2D image information and combining it with 3D scene information to obtain a 3D scene model. Thereby, we see the chance to use more sophisticated controllers for closed-loop control or online grasp planning. The segmentation network used in this work requires only a relatively small number of operations compared to state-of-the-art networks and the selected FPGA compared to other devices is [28] is rather at the low-end spectrum of performance due to design effort and space constraints. Another limitation of this work is the fact that the training of the network requires hand crafted label data and we only conduct experiments on a set of

three objects. To train the segmentation network for a larger set of objects, a more automated approach is needed, where annotation of training data could be performed offline by a larger network. In this work, we have investigated in-finger vision for the purpose of processing of visually obtained scene information. However, camera-based tactile sensors produce similar output data and we see chances of beneficial use of the methods presented in this work also in combination with tactile sensing.

In our future work, we aim at integrating additional sensor modalities in the hand including depth sensors and further refine and implement more sophisticated grasp control.

VI. Conclusion

In this paper, we have presented how visual data obtained from in-finger cameras of a soft humanoid hand can be processed in-hand and thereby provide control input for visually-guided task controllers: We show ability of fast reactive grasping, human-robot object handover and dexterous grasping of small objects.

We present the design of an in-hand hardware-accelerated convolutional neural network for object segmentation implemented on the internal FPGA of the KIT Finger-Vision Soft Hand. We present a custom energy efficient and task specialized image processing hardware design that allows inference of the encoder-decoder CNN accounting for 16.77 MOP in $5.46\,\mathrm{ms}$ and thereby satisfies real-time constraints.

We evaluate our system on a set of real-world experiments where we use the hand attached to the humanoid robot ARMAR-6 and achieve encouraging results. Comparing the response time during the ruler catching task to human performance, the median system response time outperforms all 4 test persons. We test reliability during a handover task, where 9 out of 10 repetitions proceeded successfully and demonstrate the ability of dexterous grasping using a two finger pinch grasp to lift a pencil out of a cup.

We show, that our presented design can be an alternative to widely used static or head-mounted camera setups for scene perception and show chances of reactive manipulation relying only on visual data from inside the fingers. In our opinion, the use of in-finger visual data shows promising future potential and opportunities to contribute to more versatile and improved methods for fast, reliable and dexterous robotic grasping and manipulation.

References

[1] G. Du, K. Wang, S. Lian, and K. Zhao, "Vision-based robotic grasp detection from object localization, object pose estimation to grasp estimation: A review," *arXiv preprint arXiv:1905.06658*, 2019.

[2] D. Kragic, H. I. Christensen, *et al.*, "Survey on visual servoing for manipulation," *Computational Vision and Active Perception Laboratory, Fiskartorpsv*, vol. 15, p. 2002, 2002.

[3] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[4] P. Weiner, J. Starke, F. Hundhausen, J. Beil, and T. Asfour, "The kit prosthetic hand: design and control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3328–3334, IEEE, 2018.

[5] M. Quigley, C. Salisbury, A. Y. Ng, and J. K. Salisbury, "Mechatronic design of an integrated robotic hand," *The International Journal of Robotics Research*, vol. 33, no. 5, pp. 706–720, 2014.

[6] A. Yamaguchi and C. G. Atkeson, "Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 1045–1051, IEEE, 2016.

[7] K. Shimonomura, H. Nakashima, and K. Nozu, "Robotic grasp control with high-resolution combined tactile and proximity sensing," in *2016 IEEE International Conference on Robotics and automation (ICRA)*, pp. 138–143, IEEE, 2016.

[8] M. Sato, A. Takahashi, and A. Namiki, "High-speed catching by multi-vision robot hand," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9131–9136, 2020.

[9] F. Hundhausen, J. Starke, and T. Asfour, "A soft humanoid hand with in-finger visual perception," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8722–8728, 2020.

[10] T. Asfour, M. Waechter, L. Kaul, S. Rader, P. Weiner, S. Ottenhaus, R. Grimm, Y. Zhou, M. Grotz, and F. Paus, "Armar-6: a high-performance humanoid for human-robot collaboration in real-world scenarios," *IEEE Robotics & Automation Magazine*, vol. 26, no. 4, pp. 108–121, 2019.

[11] A. Schmitz, M. Maggiali, L. Natale, B. Bonino, and G. Metta, "A tactile sensor for the fingertips of the humanoid robot icub," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2212–2217, IEEE, 2010.

[12] J. A. Fishel, V. J. Santos, and G. E. Loeb, "A robust micro-vibration sensor for biomimetic fingertips," in *2008 2nd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*, pp. 659–663, IEEE, 2008.

[13] V. Wall and O. Brock, "Multi-task sensorization of soft actuators using prior knowledge," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9416–9421, IEEE, 2019.

[14] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez, "Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1927–1934, IEEE, 2018.

[15] G. Zöller, V. Wall, and O. Brock, "Active acoustic contact sensing for soft pneumatic actuators," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2020.

[16] J. Segil, R. Patel, J. Klingner, R. F. ff Weir, and N. Correll, "Multimodal prosthetic fingertip sensor with proximity, contact, and force localization capabilities," *Advances in Mechanical Engineering*, vol. 11, no. 4, 2019.

[17] A. Yamaguchi and C. G. Atkeson, "Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 1045–1051, IEEE, 2016.

[18] B. Fang, F. Sun, C. Yang, H. Xue, W. Chen, C. Zhang, D. Guo, and H. Liu, "A dual-modal vision-based tactile sensor for robotic hand grasping," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4740–4745, IEEE, 2018.

[19] T. Lampe and M. Riedmiller, "Acquiring visual servoing reaching and grasping skills using neural reinforcement learning," in *The 2013 international joint conference on neural networks (IJCNN)*, pp. 1–8, IEEE, 2013.

[20] M. Yan, I. Frosio, S. Tyree, and J. Kautz, "Sim-to-real transfer of accurate grasping with eye-in-hand observations and continuous control," *arXiv preprint arXiv:1712.03303*, 2017.

[21] U. Viereck, A. Pas, K. Saenko, and R. Platt, "Learning a visuomotor controller for real world robotic grasping using simulated depth images," in *Conference on Robot Learning*, pp. 291–300, PMLR, 2017.

[22] D. Morrison, P. Corke, and J. Leitner, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," *arXiv preprint arXiv:1804.05172*, 2018.

[23] B. Bäuml, O. Birbach, T. Wimböck, U. Frese, A. Dietrich, and G. Hirzinger, "Catching flying balls with a mobile humanoid: System overview and design considerations," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pp. 513–520, IEEE, 2011.

[24] K. Koyama, K. Murakami, T. Senoo, M. Shimojo, and M. Ishikawa, "High-speed, small-deformation catching of soft objects based on active vision and proximity sensing," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 578–585, 2019.

[25] Z. Wan, B. Yu, T. Y. Li, J. Tang, Y. Zhu, Y. Wang, A. Raychowdhury,

and S. Liu, "A survey of fpga-based robotic computing," *arXiv preprint arXiv:2009.06034*, 2020.

[26] H. Liu, P. Meusel, G. Hirzinger, M. Jin, Y. Liu, and Z. Xie, "The modular multisensory dlr-hit-hand: Hardware and software architecture," *IEEE/ASME Transactions on mechatronics*, vol. 13, no. 4, pp. 461–469, 2008.

[27] L. B. Bridgwater, C. Ihrke, M. A. Diftler, M. E. Abdallah, N. A. Radford, J. Rogers, S. Yayathi, R. S. Askew, and D. M. Linn, "The robonaut 2 hand-designed to do work with tools," in *2012 IEEE International Conference on Robotics and Automation*, pp. 3425–3430, IEEE, 2012.

[28] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, "A survey of fpga-based neural network accelerator," *arXiv preprint arXiv:1712.08934*, 2017.

[29] L. Lu, Y. Liang, Q. Xiao, and S. Yan, "Evaluating fast algorithms for convolutional neural networks on fpgas," in *2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp. 101–108, IEEE, 2017.

[30] S. Anwar, K. Hwang, and W. Sung, "Fixed point optimization of deep convolutional neural networks for object recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1131–1135, IEEE, 2015.

[31] D. Lin, S. Talathi, and S. Annapureddy, "Fixed point quantization of deep convolutional networks," in *International conference on machine learning*, pp. 2849–2858, PMLR, 2016.

[32] L. Lai, N. Suda, and V. Chandra, "Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus," *arXiv preprint arXiv:1801.06601*, 2018.

[33] F. Hundhausen, D. Megerle, and T. Asfour, "Resource-aware object classification and segmentation for semi-autonomous grasping with prosthetic hands," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, pp. 215–221, IEEE, 2019.

[34] N. Vahrenkamp, M. Wächter, M. Kröhnert, K. Welke, and T. Asfour, "The robot software framework armarx.," *it Inf. Technol.*, vol. 57, no. 2, pp. 99–111, 2015.

[35] G. Del Rossi, A. Malaguti, and S. Del Rossi, "Practice effects associated with repeated assessment of a clinical test of reaction time," *Journal of athletic training*, vol. 49, no. 3, pp. 356–359, 2014.