Grasping with Humanoid Hands based on In-Hand Vision and Hardware-accelerated CNNs

Felix Hundhausen, Simon Hubschneider and Tamim Asfour

***Abstract*— We present a vision-based grasping system for humanoid and prosthetic hands using hardware-accelerated CNNs for real-time object classification and class-aware pixel-wise segmentation. The system is implemented on the hand internal processing hardware of a humanoid hand using a System-on-Chip (SoC) comprising a Processor System (PS) and an FPGA. As a sensor system, the hand provides an integrated RGB camera, a multi-region Time-of-Flight (ToF) depth sensor, and an Inertial Measurement Unit (IMU). We propose an algorithm for 3D object shape estimation based on sensory information provided by the hand internal sensor system. The 3D object mesh in combination with the object relative pose of the hand is used as input for a reactive grasp controller. For the design of the CNN-based object recognition and segmentation networks, we use a resource-aware algorithm for Network-Architecture-Synthesis (NAS). We evaluate the visual perception accuracy and 3D model estimation accuracy in grasping experiments with six objects. We obtain a mean object segmentation accuracy of 84.4 % and a mean error for object diameter estimation of 44 mm.**

I. Introduction

The development of highly integrated humanoid and prosthetic hands that are versatile in terms of their grasping abilities remains a challenging problem. In our research, we aim at developing anthropomorphic hands that are equipped with intelligent functionalities to improve autonomous grasping in humanoid robotics and semi-autonomous grasping in prosthetics [1], [2]. To achieve this, we combine under-actuation mechanisms for the mechanical design with in-hand integrated multimodal sensor systems and embedded systems to improve sensory-based grasping. In particular, we developed resource-aware CNN-based solutions for the processing of visual information obtained from in-hand integrated cameras [3], [4]. In this paper, we present a novel 3D aware grasping system for humanoid and prosthetic hands that should support intelligent control of such hands in grasping. The advantages of in-hand vision over hand external cameras are a better view of the object without occlusion by the hand or other objects, further precise grasping is possible without a precise estimation of the hand pose. To not rely on hand external computing systems and a high bandwidth data-interface it is beneficial to realize the complete computation hand internal, however this comes with the challenge to design algorithms, that are suitable for resource-constraint processing system but still can fulfill reactive grasping as expected by the users. Recent machine learning methods that are well suited for visual scene perception require a large number of compute operations. To overcome this challenge, hardware-accelerated processing can be used, here FPGAs allow the flexible implementation of highly task-optimized processing architectures.

This work has been supported by the Carl Zeiss Foundation through the JuBot project.

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

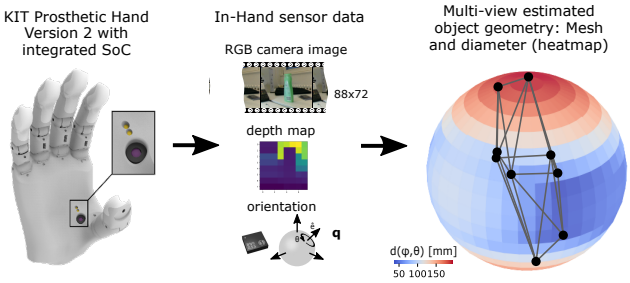


Fig. 1: Overview of the proposed system: The in-hand sensor data of the KIT Prosthetic Hand is used for in-hand multi-view object mesh estimation to allow estimation of the orientation-specific object diameter (heatmap)

In this work, we present a system for real-time in-hand 3D visual scene perception for grasping with humanoid hands. Hereby, we focus on combining multiple views and multi-modal sensory information to generate an object mesh and estimate the object diameter which is then used as the input for a grasping controller.

Contributions: *(i)* An FPGA-accelerated in-hand processing system for real-time visual object classification and segmentation using CNNs. We design the network architectures using resource-aware Network Architecture Search (NAS) as described in Sec. V-A. *(ii)* A multi-view object shape estimation using the obtained binary object mask, inertial sensor data from an IMU and depth information from a Time-of-Flight (ToF) sensor (Sec. V-B). To this end, we adapt the CNN hardware implementation from our previous work [5] for layer-wise CNN acceleration that allows flexible inference of different CNN architectures as required for classification and pixel-wise image segmentation tasks. As a result, we present a completely hand-integrated system for 3D scene-aware grasping without the need for any external high-power computing resources. The system can estimate 3D object shape based only on object classification and segmentation, without the need for 3D object models. *(iii)* We evaluate the perception system in terms of performance and accuracy on recorded data and evaluate the complete system by grasping a set of objects based on the obtained 3D object mesh.

II. Related Work

Reliable grasping and manipulation of objects with humanoid hands is still a challenging task. For successful grasping, information about the object is needed to plan and execute a grasp. This includes information about the object geometry, the pose of the object relative to the hand and suitable preshapes of the hand, i. e., the configuration of the fingers. When using hand internal vision, small errors in the estimated object pose or the hand pose lead to significant pose errors of the object in relation to the hand. A solution to this problem is the use of in-hand cameras that can directly estimate the object pose in relation to the hand. Numerous approaches from the field of robotics use camera-in hand approaches, where a camera is attached to a gripper that is interfaced with an external computer via a high bandwidth data connection. This leads to complex and bulky hardware setups, which should be avoided in a humanoid robot. This problem could be solved by grippers or humanoid hands with a real-time capable embedded sensor-processing system. In addition to the use case of robotics, these hands would also provide helpful functions for the development of prostheses with semi-autonomous grasping abilities, since many recently published systems rely on external perception and data processing hardware. In our previous work, we have evaluated the use of microcontroller-based hand internal processing for visual object classification[6]. Further, we have designed hardware and implemented methods for in-hand hardware-accelerated object perception that allows image-based reactive grasping [5].

To achieve the best possible network accuracy with constrained hardware, Network Architecture Synthesis allows to obtain optimized network architectures. Here three major aspects need to be considered: Description of the search space, search strategy, and the selected performance estimation method [7]. In the case of neural networks, the search space is described by a graph of elementary blocks, which are either single network layers or more complex units such as skip-connections [8]. Based on the search space, a network architecture can be obtained by using different optimization methods. However, all these methods require the evaluation of multiple different solution candidates, which in some cases necessitate training only a subset of the available data or for fewer epochs to speed up the accuracy estimation. In resource-aware approaches, the objective function is more complex and may also include parameters like the energy needed for execution [9], inference latency [9], [10] and the number of multiply-accumulation (MAC) operations [8], [6]. These parameters can be treated both as optimization goals [9] or constraints [10], [8], [6] and are highly problem-specific.

For obtaining 3D object models from annotated visual input that can be used for grasp control, the model can be reconstructed from multiple views with additional depth information from a Time-of-Flight (ToF) depth sensor and orientation information. 3D reconstruction is a topic of high interest in computer vision and graphics [11] and is also very relevant for mobile robotics in general. However, our focus is not primarily on a high quality of the reconstructed geometric models, but the pose of the robot in combination with a more sparse scene model generated in real-time.

In literature, a large set of distinguishing methods for 3D reconstruction methods can be found, these also differ depending on the types of input data like single RGB image, RGB-D image, multi-view images or video data[12]. When using multi-view data, machine learning methods or 3D scene reasoning can be used to obtain the 3D model. For mobile robotics, SLAM can provide a scene model while also giving an estimation of the robot's pose. If the pose is known from non visual sensors (odometry, kinematic chain, IMU or others), the problem reduces to obtaining the scene model.

An overview of methods for obtaining occupancy-based object descriptions using voxels by shape-from-silhouette techniques is given in [13]. These methods take segmented RGB-images or depth images as input. Other approaches do not rely on the volumetric grid and aim at fitting the estimated object-volume by low-parameter models like ellipsoids or quadrics [14] [15] [16].

In this work, we use RGB data in combination with low-resolution depth data from a ToF-depth sensor to find object coordinate points and to obtain a low-resolution object mesh. This allows including higher uncertainty points and allows simple fusion of multiple perspectives using mesh optimization.

III. System overview

In this work, we present methods for semi-autonomous grasping using multi-modal sensory input from a hand-integrated sensor setup inside the humanoid hand. Our goal is to endow humanoid robot hands and hand prostheses with more autonomous grasping abilities.

Our system initially performs object classification on the RGB camera input data stream. The RGB image is downsampled and slightly cropped to match the dimensions of (88×72 pixels). The hand internal processing system controls the inference of the classification architecture. The result is shown on the display. As soon as the robot or user starts the grasp by accepting the detected object class, the object model generation in triggered. In our prototype, the start can be triggered by a push-button. For the model generation, the camera image is segmented by an encoder-decoder CNN which outputs a pixel-wise binary object segmentation mask. From the binary mask, the object principle axes are calculated and back-projected to 3D based on the IMU and depth information. When multiple views from different orientations are collected, the model generation and grasp controller can be triggered. Thereby the grasp aperture is selected according to the current hand orientation in relation to the object. A graphical overview is is given in Fig. 2.

IV. Hardware platform

For the experiments in this work, we use the second version of the KIT Prosthetic Hand (50^{th} percentile female)

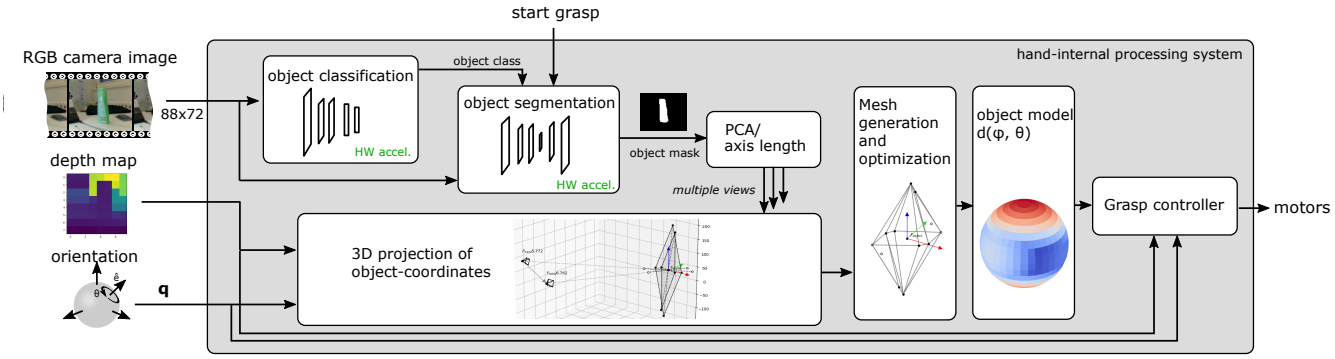


Fig. 2: The proposed system: The recognized object is segmented pixel-wise in multiple camera views from different poses, and the obtained object mask is used to estimate the object axis dimensions. Object coordinate points from the image plane are back-projected to 3D using the hand pose relative to the object. After mesh generation and optimization, the object diameter $d(\varphi, \theta)$ can be estimated and used for grasp control.

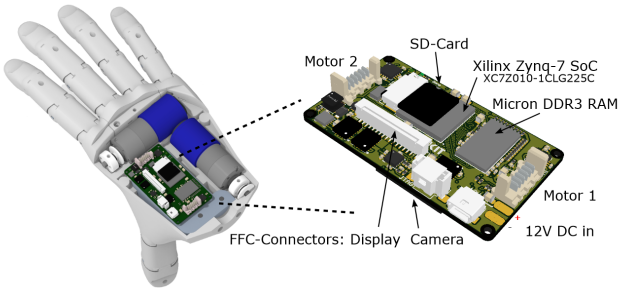


Fig. 3: KIT Prosthetic Hand V2 with our control hardware based on a Xilinx Zynq System-on-Chip (SoC) including a dual-core processing system (PS) and programmable logic (PL)

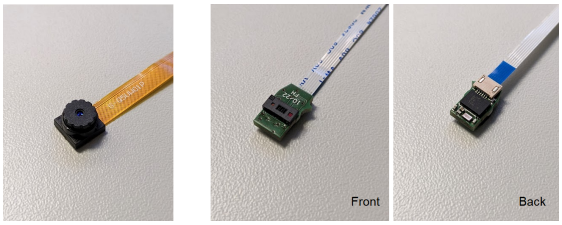


Fig. 4: Hand integrated miniature sensor-setup: 5 MP RGB-camera (left) and $11.6\,\mathrm{mm} \times 8.2\,\mathrm{mm}$ sensor PCB with ToF depth sensor on the front and IMU chip on the back side of the PCB (right).

([17]) with an internal Xilinx Zynq Z7010 SoC-based control hardware as introduced in [18]. The hand has two DC gear motors for tendon-based actuation, which include relative encoders for position control of the thumb and the other four coupled fingers. The mechanism for underactuation is described in [17]. The complete control and data processing hardware is realized on a hand internal PCB, the data processing system consists of a dual-core processor and reconfigurable hardware (FPGA) on a Xilinx Zynq-7 SoC (XC7Z010-1CLG225C). Further, the hand includes a miniaturized and completely integrated sensor system consisting of a miniature camera (OmniVision OV5640), a multi-zone time-of-flight distance sensor (ST VL53L5CX), as well as an IMU (Bosch Sensortec BNO055) that allows the hand internal estimation of the orientation. The set of sensors is shown in Fig. 4. The user can obtain feedback from a display on the back side of the hand.

V. Approach

In the following, we describe the two subsystems of CNN-based object perception and object shape estimation.

A. CNN-based Object perception

For the realization of the object perception system, we record training data that we use to optimize the network architecture and train the final architecture that can be executed on the hardware accelerator.

1) Dataset: The dataset for the training of the CNNs for object classification and segmentation was recorded with the hand internal RGB camera. The data set is recorded with 6 different objects from YCB[1][19] and KIT[2][20] object dataset. (banana[1], spam[1], showergel[2], pitcher[1], bowl[1], hammer[1]). We recorded the images in a slightly cluttered office/desktop environment with different illumination conditions and varying object poses. In total ≈1500 images were recorded and annotated with a pixel-wise ground-truth mask. An exemplary image (showergel) is shown in Fig. 6. As a test set, we separately record RGB data during the grasping experiments (see Sec. VII) and annotate these images with ground-truth object masks.

2) CNN architecture synthesis: The goal of the Network-Architecture-Synthesis (NAS) is to achieve the highest possible classification and segmentation accuracy while still taking real-time requirements and the constraints of hand internal data processing hardware and the accelerator into account. We employ a genetic evolutionary approach that co-optimizes accuracy, inference latency, and the number of feature maps, which are the primary driver for FPGA resource utilization. The network architectures are encoded as a sequence of basic layers: convolution, max-pooling, and upsampling with parameters as supported by the hardware implementation. To ensure proper output formats, dense layers for object classification are added independent of this

encoding, and the segmentation masks are created by two convolution layers with prior upsampling to the input resolution if needed. We reject any solution candidate that violates a maximum inference time of 100 ms, the upper feature map size limit of 88×72 and the lower limit of 11×9 pixels. Our approach uses tournament selection to select the best individuals from a given population of solution candidates, which then serve as parents to new solutions created by one-point crossover of their layer sequence. Lastly, the combined population is reduced back to the original size by tournament selection in a process called environmental selection. From a randomly sampled initial population, this evolutionary cycle is repeated for a fixed number of generations.

Since we formulate the NAS as a multi-objective optimization problem, the solution is a set of pareto-optimal candidates. Figure 5 exemplary shows the 12 members of this pareto-front (✖) of a search for classification architectures, annotated by color with the number of feature maps they require. The plot includes also investigated solutions that are not part of the pareto front (●).

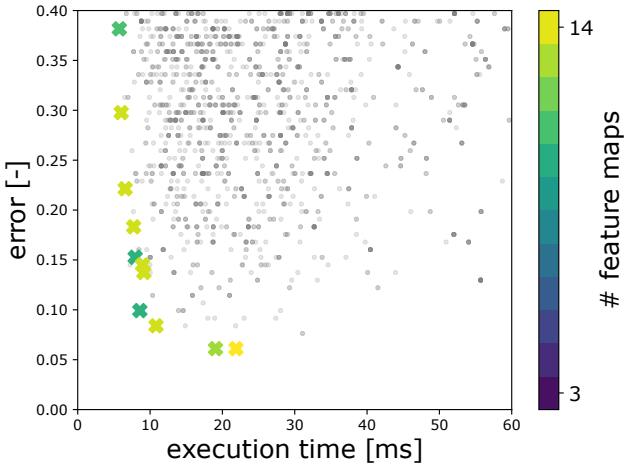


Fig. 5: Resulting pareto front of a classification network architecture search next to the sub-optimal solution candidates evaluated during search.

3) FPGA based hardware accelerator: For inference of the perception CNNs we use a CNN hardware-accelerator implemented on the Programmable Logic (PL) of the hand-integrated FPGA. The accelerator hardware supports convolution, max-pooling (2×2) and upsampling (2×2) layers. Especially convolution layers require a large number of multiply-accumulate (MAC) operations and benefit from parallel computation on the FPGA. The number of the hand internal FPGA's provided DSP-blocks allows the implementation of 14 parallel convolution units with a 3×3 kernel size that enables parallel processing of 14 output feature maps. Dense layers that are typically used to obtain the output vector do not allow such a speedup and are executed by the processing core.

In our previous work [5] we have used the CNN accelerator only for real-time image segmentation. In this work, we implemented a control interface to the processing system that allows flexible inference of different network architectures controlled by accelerator code, including segmentation and classification network architectures. The network weights are trained offline and accelerator code and quantized weights are generated and included in firmware to be executed by the processing system with the use of the FPGAs accelerator hardware. The accelerator configuration is transmitted by a serial interface from PS to PL and the input data is transferred via the AXI bus into the BRAM of the PL. The accelerator output data can be read out in the same way. The output of the segmentation network consists of two feature maps, one indicating the object and the other one indicating the background. The binary output image is obtained using a greater-than operation.

B. *Object shape estimation*

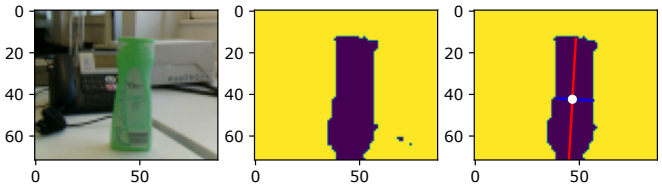


Fig. 6: Example image from the hand internal camera (left) and binary object mask obtained from the segmentation CNN (center). We use morphological operations to reduce noise and PCA to determine the orientation of the principle axes (right) and measure the length of these axes to obtain four object coordinate points.

To reduce noise in the binary output image, we apply two erosion and two dilation operations to the raw output image and estimate the object center by calculating the centroid of all positively segmented pixels. To exclude views where the object is only partially visible, we detect object pixels at the image border and reject these images during processing. In the following step, eigenvalues and eigenvectors of the segmentation mask are calculated to estimate the object's principal axis. The length of the object axis is detected by pixel-wise sampling along the two axes. These steps are shown in Fig. 6. For each view, we use the 4 axis endpoints as *object coordinate points* (●). Two further points are estimated normal to the image plane, the length of this axis is estimated with high uncertainty (○) as the mean value of the other two axis lengths. If the object is only partially visible, a high uncertainty point at the corresponding axis endpoint is registered.

The distance between hand camera and object is determined using depth information from the ToF sensor. The distance value is selected from the 8×8 depth image according to the object center coordinate. It is then used to estimate the object dimensions in millimeter from the length measured in the image plane. The hand orientation is obtained from the integrated IMU. We define the object's center as the coordinate system's origin and assume that the scene is static and the object is not moving. The camera/hand pose is obtained using the distance value, the hand orientation and

the position of the object in the image. This allows projecting the object's coordinate points to 3D space. To avoid highly redundant data, we only record object coordinate points in case the hand's angular position from the previous pose is larger than $\frac{\pi}{8}$.

To fuse geometric information from multiple views, we aim to find a convex polygon mesh with maximum volume where mesh vertices can be ● -points and ○ -points. ● -points are not allowed to lie inside the mesh volume (since these points are detected as object surface points), while ○ -points are allowed to lie inside the mesh volume. We use a heuristic to find such a set of points and triangle surfaces. We generate an initial object mesh by connecting all coordinate points (○ and ●) into a triangle mesh with minimal edge length. To find a mesh that only consists of ● -points, for each point we calculate

1) By how many faces is the point excluded from a convex volume.
2) How often does a face adjacent to this point exclude another point.

If the constraints are not met, the following steps are executed:

- Flip edges if this reduces the total excluded point score.
- If any ○ point does exclude other points (2)), these points are excluded from the mesh.
- Remove ● -points that are excluded the most often (1).

Fig. 7 visualizes the result of the algorithms with two initial views and the corresponding coordinate points. Further the mesh before and after optimization including the scores (1)) and (2)) are annotated in red and magenta.

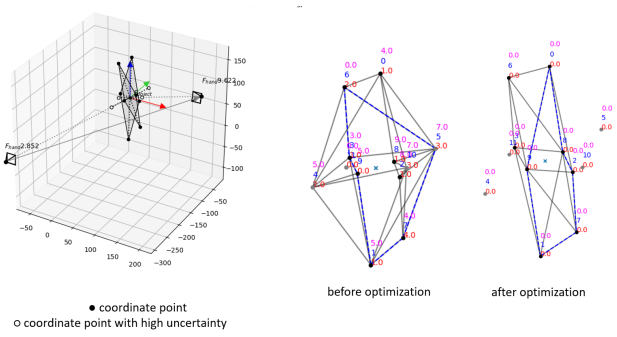


Fig. 7: Example of mesh generation and optimization from two views. (showergel-object)

VI. Grasp Controller

The grasp controller controls finger joint angles depending on hand-object orientation specific object diameter. The controller obtains the generated object mesh and uses IMU orientation data to calculate the diameter depending on hand-object orientation. ($d(\varphi, \theta)$). Depending on hand-object distance that is obtained from the depth data, the controller either sets the position controlled finger joint angles for a pre-grasp pose or executes the actual grasp. For the pre-grasp the finger aperture is set 30% larger than the actual object diameter. The actual grasp is started when hand and object are in close proximity. ($D < D_{th}(grasp)$) The finger position is here set to the actual object diameter. As soon as the finger position is reached, the motors are set to constant voltage to apply force to the object. For $D_{th}(grasp)$ we have evaluated $60\,\mathrm{mm}$ as an optimal value for fluid grasp execution.

VII. Experimental Evaluation

To evaluate the methods for object detection and shape estimation, we conduct an experiment where we use the hand to grasp a set of 6 objects. We attach a handle for handheld use of the hand. For power supply, a $12\,\mathrm{V}$ DC voltage source is connected and the data from sensors and the obtained results are saved to the hand internal micro SD-card. The command to start the grasping process is given by the user by pressing a push-button integrated into the handle.

In the experiment, the user grasps 6 objects from the YCB[1][19] and KIT[2][20] object dataset: banana[1], spam[1], showergel[2], pitcher[1], bowl[1], hammer[1]. The ground-truth data is obtained from 3D models included in the databases. During the experiments, the objects are placed in a random orientation on a table in a typical office/desktop environment, with the typical orientation of the object (pitcher standing on bottom side, e.g.). During the grasp, the user points the hand camera in direction of the object, so that one up to three (but mostly two) object views from different perspectives are recorded.

We evaluate the accuracy of the perception subsystem including the classification and segmentation network's accuracy and evaluate results of the object shape estimation based on hand internal sensor data.

A. Evaluation of CNN based object perception

The architecture for the classification CNN consists of 6 convolution layers with and $2\times$ 2 pooling after the first two convolution layers and twice after the third. The number of convolution output feature maps is 9, 14, 6, 14, 9, 6. The last two layers are dense layers with 32 and 6 output units.

For the classification network we obtain an accuracy of 94.89% on the test set consisting of image data recorded during the grasping experiments. The total number of Multiply-Accumulate-Operations are 10.1 MOP. The run-time of the classification network on the hardware accelerator was measured as $29.2\,\mathrm{ms}$ resulting in 346 MOPS and $34.2 fps$. The obtained run-time is well above the frame rate of the camera ($10.4 fps$) and allows online classification of camera images which enables a reactive grasping process and allows e. g.., the user to online observe the detected object class.

To evaluate the accuracy of the object segmentation during the experiments, we record the camera RGB images during the grasping experiments and evaluate the segmentation accuracy by comparing the result to hand annotated ground truth object masks. We calculate the value for the Intersection over Union (IoU) using the number of true positive (TP), false positive (FP) and false negative (FN) segmented pixels as follows:

$$IoU = \frac{TP}{TP + FP + FN} \tag{1}$$

The weights for the object segmentation network are trained class-wise. Here, we evaluate the mean value for IoU for each object as shown in Fig. 8.

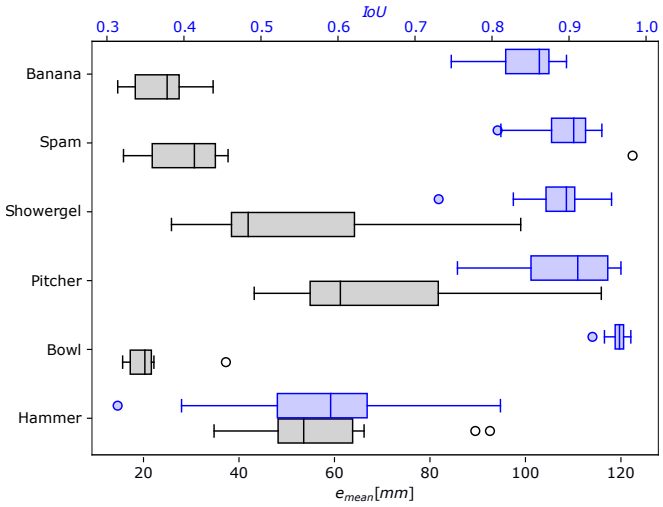


Fig. 8: Boxplot of obtained segmentation accuracy (IoU) and e_{mean} of the obtained mesh from the combined views. 10 grasping trials per object class with mostly 2 views (± 1) per class were recorded and compared to ground truth object segmentation and ground truth object models.

As mean values for each object for pixel-wise segmentation we obtain the following values: Banana: 0.85, Spam: 0.89, Showergel: 0.88, Pitcher:0.89, Bowl: 0.96, Hammer: 0.58.

The selected architecture for segmentation has the following number of feature maps: 8, 8, 8, 8, 1, 1. One 2×2-pooling layer is executed after the first convolution layer. The total number of MAC-operations for the segmentation architecture are 7.08 MOP. The run-time of the segmentation network on the hardware-accelerator was measured as $16.8\,\mathrm{ms}$, resulting in 421 MOPS. The achieved run-time can satisfy real-time requirements since every camera frame can be segmented in a very fast way, leaving sufficient time ($79.4\,\mathrm{ms}$ per frame) for detection of object dimensions and registration of coordinate points.

B. Evaluation of object shape estimation

During the grasp experiments we record the estimated object meshes and manually compare these to ground truth data for the 6 object classes. The ground truth data point clouds are manually positioned with help of the captured RGB images and the recorded hand poses. Like for the grasp controller, we calculate the object specific diameter $d(\varphi,\theta)$ based on the estimated object mesh and the ground truth point cloud. The error between estimated model and ground truth $e(\varphi,\theta)$ is calculated according to equation 2.

$$e(\varphi,\theta) = |d(\varphi,\theta) - d_{GT}(\varphi,\theta)| \tag{2}$$

To obtain the mean error e_{mean}, we integrate $e(\varphi,\theta)$ over the unit sphere and divide it by the sphere's surface area S according to equation 3.

$$e_{mean} = \oiint_S e(\varphi,\theta)\,d\varphi\,d\theta/S \tag{3}$$

The results for e_{mean} obtained in 10 trails for the 6 objects are shown in Fig. 8. As mean values for all grasp trial, we obtain an e_{mean} as: Banana: $24.14\,\mathrm{mm}$; Showergel: $51.61\,\mathrm{mm}$; Spam: $37.15\,\mathrm{mm}$; Pitcher: $69.74\,\mathrm{mm}$; Bowl: $21.10\,\mathrm{mm}$; Hammer: $58.91\,\mathrm{mm}$. One exemplary mesh estimation result for each object and the obtained value for e_{mean} is visualized in Fig. 9. Best accuracies are obtained for the bowl and banana object, here the objects are lying on a flat table which allows for a more accurate measurment of the object distance. If the background is further away from the object, in some cases depth estimation is not accurate due to noisy and faulty matching of depth values . This results in a wrong estimation of object size and we see this as a reason for higher errors for the pitcher and the showergel object.

The algorithm for mesh generation and optimization is executed on the embedded ARM Cortex-A9 processor running at $f_{CPU} = 400\,\mathrm{MHz}$. We evaluate the run-time of the algorithm for up to 5 views recorded while approaching the YCB-bowl for grasping. The results are shown in Tab. I. It can be seen that for an increasing number of views, the run-time drastically increases, while the generated mesh does not provide a significantly better resolution. We see two or three views as a suitable number since also in practical use each recording of a new view would take additional time.

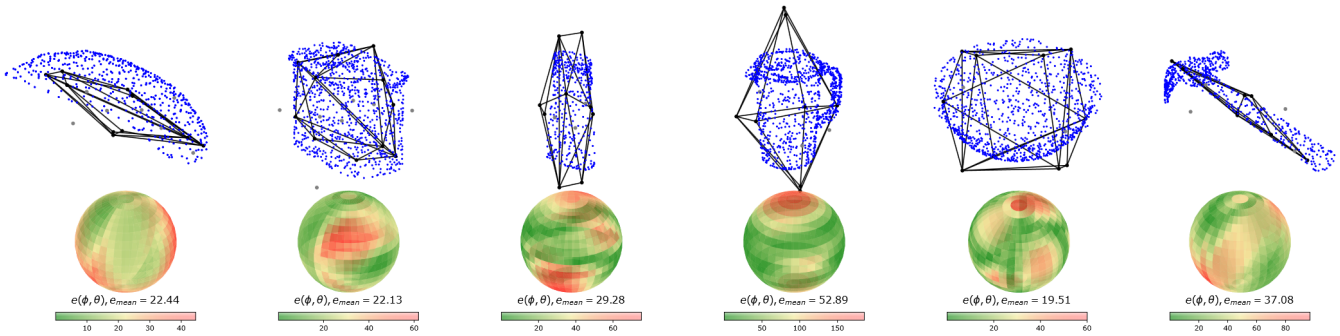
Views	**Points**	**Mesh triangles** (after optimization)	**Run-time**
1	6	8	0.3 ms
2	12	20	4.4 ms
3	18	18	134 ms
4	24	20	224 ms
5	30	24	588 ms

TABLE I: Run-time of the mesh generation and optimization algorithm

VIII. Discussion and Conclusion

In this paper we have presented methods for visual scene perception and generation of 3D object models that enable grasping with a humanoid hand. We presented a hardware system for accelerated CNN processing in combination with an algorithm for resource-aware network architecture synthesis. The combination of real-time-capable classification and segmentation CNNs allows the implementation of reactive grasping behaviors. We presented a real-time capable algorithm for obtaining the mesh from multiple object coordinate points obtained in 3D space using the hand internal IMU and depth information. In our evaluation we investigate classification, segmentation and mesh generation accuracy and in the conducted experiment we can reliably grasp all of the 6 objects.

Up to now, we have only evaluated our methods on a limited number of objects with a handheld humanoid hand. In this work we have used methods to approximate the object shape by a convex mesh, which allows good approximation of many objects in our test set. However, for more complex



((a)) Banana ((b)) Spam ((c)) Showergel ((d)) Pitcher ((e)) Bowl ((f)) Hammer

Fig. 9: Visualization of ground-truth point cloud in blue and the estimated object model mesh in black. The heatmap-sphere shows the orientation specific error $e(\varphi, \theta)$ in mm as given by equation (2). The mean error e_{mean} is calculated according to equation (3).

object shapes the proposed method is not well suited. In future work we aim at more detailed model generation to reduce the shape estimation error rates.

For training of new objects a relatively high effort is required to annotate the visual data. However, for offline training of the CNN, larger pre-trained models could be used to reduce the effort of manual labeling. In our current set of objects, especially objects with a non uniform color are segmented with higher error rates and thus the accuracy of the obtained mesh is lower. Thus, a more powerful processing hardware compared to the currently used Zynq 7010 can provide higher accuracy but would also require larger datasets for training the network's weights. In addition, we have used object coordinate points obtained at principal axis in combination with mesh generation and optimization. However, other methods for object pose estimation based on models that can handle perception uncertainty and video data could probably provide higher accuracy, especially for objects with a non-trivial geometry. However, the system investigated in this work allows reliable grasping of all evaluated objects in a reactive way.

References

[1] P. Weiner, J. Starke, S. Rader, F. Hundhausen, and T. Asfour, "Designing prosthetic hands with embodied intelligence: The kit prosthetic hands," *Frontiers in Neurorobotics*, vol. 16, pp. 1–14, 2022.

[2] T. Asfour, M. Wächter, 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.

[3] F. Hundhausen, R. Grimm, L. Stieber, and T. Asfour, "Fast reactive grasping with in-finger vision and in-hand fpga-accelerated cnns," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 0–0, 2021.

[4] 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.

[5] F. Hundhausen, R. Grimm, L. Stieber, and T. Asfour, "Fast reactive grasping with in-finger vision and in-hand fpga-accelerated cnns," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6825–6832, IEEE, 2021.

[6] 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.

[7] T. Elsken, J. H. Metzen, and F. Hutter, "Neural Architecture Search: A Survey," Apr. 2019. arXiv:1808.05377 [cs, stat].

[8] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically Designing CNN Architectures Using the Genetic Algorithm for Image Classification," *IEEE Transactions on Cybernetics*, vol. 50, pp. 3840–3854, Sept. 2020. Conference Name: IEEE Transactions on Cybernetics.

[9] L. Cai, A.-M. Barneche, A. Herbout, C. S. Foo, J. Lin, V. R. Chandrasekhar, and M. M. Sabry Aly, "TEA-DNN: the Quest for Time-Energy-Accuracy Co-optimized Deep Neural Networks," in *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 1–6, July 2019.

[10] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "MnasNet: Platform-Aware Neural Architecture Search for Mobile," May 2019. arXiv:1807.11626 [cs].

[11] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb, "State of the art on 3d reconstruction with rgb-d cameras," in *Computer graphics forum*, vol. 37, pp. 625–652, Wiley Online Library, 2018.

[12] H. Ham, J. Wesley, and H. Hendra, "Computer vision based 3d reconstruction: A review," *International Journal of Electrical and Computer Engineering*, vol. 9, no. 4, p. 2394, 2019.

[13] C. R. Dyer, "Volumetric scene reconstruction from multiple views," *Foundations of image understanding*, pp. 469–489, 2001.

[14] V. Gaudillière, L. Pauly, A. Rathinam, A. G. Sanchez, M. A. Musallam, and D. Aouada, "3d-aware object localization using gaussian implicit occupancy function," *arXiv preprint arXiv:2303.02058*, 2023.

[15] C. Dune, E. Marchand, C. Collowet, and C. Leroux, "Active rough shape estimation of unknown objects," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3622–3627, IEEE, 2008.

[16] C. Rubino, M. Crocco, and A. Del Bue, "3d object localisation from multi-view image detections," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1281–1294, 2017.

[17] P. Weiner, J. Starke, S. Rader, F. Hundhausen, and T. Asfour, "Designing prosthetic hands with embodied intelligence: The kit prosthetic hands," *Frontiers in Neurorobotics*, vol. 16, 2022.

[18] N. Fasfous, M.-R. Vemparala, A. Frickenstein, M. Badawy, F. Hundhausen, J. Höfer, N.-S. Nagaraja, C. Unger, H.-J. Vögel, J. Becker, *et al.*, "Binary-lorax: Low-latency runtime adaptable xnor classifier for semi-autonomous grasping with prosthetic hands," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13430–13437, IEEE, 2021.

[19] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *IEEE Robotics & Automation Magazine*, vol. 22, no. 3, pp. 36–52, 2015.

[20] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.