Recognition of objects to grasp and Neuro-Prosthesis control

Attila Fejér

PhD Thesis booklet

Supervisors:
Dr. Péter Szolgay
Dr. Jenny Benois-Pineau

Roska Tamás Doctoral School of Sciences and Technology
Pázmány Péter Catholic University
Doctoral School of Mathematics and Computer Science
University of Bordeaux

Budapest, 2022

Contents

Chapter 1

Introduction and open questions

The use of a prosthetic arm in modern days have become an accessible and useful tool for people who have lost their upper limbs due to an accident or different diseases. Technology leads to development of more comfortable and suitable equipment, helping the user in every-day life. Our vision is to further enhance the quality of the prosthetic arm using primary visual information. Our visual controlling mechanism must be suitable to wear and allow real-time processing while does not limit the mobility of the artificial limb. Carefully examining these requirements, we have decided to develop the controlling mechanism on FPGA. However, this is an open problem and there are intensive research activities in this area.

The controlling program has several individual parts running on a computer. Our goal is to accelerate the algorithms to achieve the real-time processing speed.

It is a complex computer vision, image processing and a robotic problem, because we have to find the object, which the user wants to grasp. This part is a computer vision and image processing task. We also want to control the robotic arm to help the user to grasp this object.

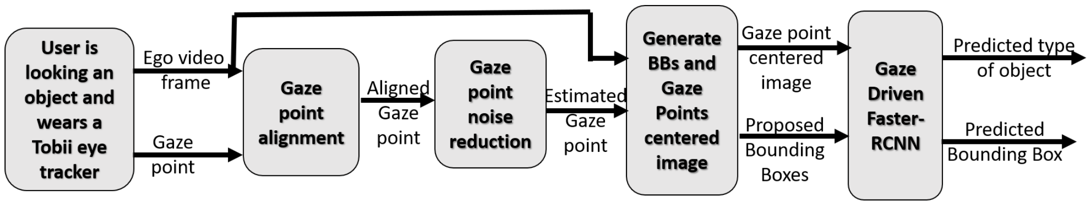

Figure 1.1: The prosthetic arm visually guided system.

To control a prosthetic arm for the amputees, we proposed a hybrid hardware-software solution. The system is in the Figure 1.1, which is showing the main components: Record the frames and gaze points with Tobii eye-tracker, gaze points alignment, gaze points noise reduction, bounding boxes generation around the noise reduced gaze points, object detection with the Gaze Driven CNN.

Tobii eye-tracker, which record the user gaze points (the point where the user is looking currently). The gaze points have 3 coordinates, x, y, gamma. Where gamma is the depth of gaze point. The Tobii eye-tracker also record an egocentric view video. The recorded video resolution is 1920 px x 1080 px (Full HD) and the framerate is 25 fps.

The next step is the gaze points alignment to extract the Scale-Invariant Feature Transform (SIFT) keypoints of every consecutive frames in a video. Our measurements show that SIFT [1] was one of the slowest part in our flow of algorithms.

SIFT is a widely used and implemented keypoint detector. There are CPU[2, 3], GPU[4, 5] and FPGA implementations[6, 7, 8, 9, 10, 11]. The OpenCV SIFT library[2] and the OpenSIFT[3] are popular frameworks for SIFT keypoint extraction and descriptor computation on a CPU. However the runtime of these CPU implementations are too slow for real-time image processing on light-weight devices. Computation can be accelereted by using GPUs, for example: CudaSIFT[5] can process a 1280px $\times$ 960px image in 12.7 ms, i.e. 78.74 frames per second (fps) on the NVIDIA GeForce GTX 580 GPU. However, its power consumption is 244W, which is too high for a wearable application. HartSIFT[4] can extract features within 3.14$\sim$10.57ms (94.61-318 fps) depending on the input image size on the NVIDIA GeForce GTX TITAN Black. The power consumption of this GPU is 250W, which is also very high for a portable device.

The most computationally intensive operation in SIFT keypoint extraction is the computation of the Gaussian pyramids as it requires multiplication of coefficients of Gaussian filters with scale-space images. For this step, an analog solution is developed by Rodríguez-Vázquez et al. [12, 13] where the Gaussian pyramid is computed by an analog CMOS circuit and exhibits very low dissipated power. The inherent parallel processing exhibits high computational power of the analog VLSI implementations. However, the size of the array is small [12, 14, 13]. In [13], the analog sensor/processor implementation has 88x60 processing elements, and each processing element has 4 photo-diodes. The computation unit is connected to the vision sensor unit of the camera. The vision sensor array has 176x120 pixels only and the system is implemented by using a 0.18 μm CMOS technology. This solution has proper processing power but the resolution of the analog sensor is not sufficiently high for our application, since our method use larger areas in video frames and it needs 480px $\times$ 480px image as input.

Several different SIFT implementations on FPGA are published such as the system designed by Doménech-Asensi et al.[6], which is a simplified version of the algorithm. It is assumed that each feature point has two main orientations at most, the orientation histogram uses 8 bins instead of 36 bins. Thus complexity is reduced but precision of orientations less with regard to the original SIFT[1]. The system can process 640px $\times$ 480px sized input images at 99 fps processing speed on Xilinx Virtex-5, and it uses fixed point representation.

Having the SIFT keypoints the next step is the FLANN matching. There is a reference image which we compare the reference image-10, reference image- . . . ., reference image-1 images and put them in the same plane. After that we can compute the homography matrix, and with that we can estimate the gaze points place in the reference frame plane.

In the gaze point noise reduction, the goal is to filter out the outlier gaze points. The outlier gaze points caused by the saccades, and the head movements, when the user try to find to object or get distracted during the process of grasping. The Density-based spatial clustering of applications with noise (DBScan) clustering removes the outlier gaze points and the Kernel Density Estimation (KDE) uses the remained aligned gaze points and estimate the location of the gaze point.

Bounding boxes generated around the fixated gaze point. Nine bounding boxes generated with different scale and size where the center point is the fixated gaze point.

The Gaze Driven CNN is predicted the object type and the object location. Inputs of this CNN is the 9 bounding boxes and the current frame. The object type and location are needed to better controlling the prosthetic arm.

Chapter 2

Research methodology

Our goal is to find a device which can control a robotic arm. It has to be low weight equipment with low power consumption. On the other hand, it should work efficiently, and the whole system has to work almost in real-time, in order to process the data quickly.

The Field Programmable Gate Arrays (FPGA) has been chosen, because it has low energy dissipation therefore, it is good for prototyping a wearable device. In that case, the user wanted to wear our application, so a light-weight device is mandatory. The FPGA has the required computational power to accelerate the chain of algorithms to make it real-time processing. The real-time processing is required in this case, since the goal of this research to use the system in every-day life.

The Xilinx Xilinx Zynq UltraScale+ MPSoC ZCU102 [15] has been used.

The XCZU9EG FPGA device on the ZCU102 board has a Processing System (PS) and a Programmable Logic (PL) part. The PS part has a quad-core Arm Cortex-A53, dual-core Cortex-R5F real-time processors, and a Mali-400 MP2 graphics processing unit. The Cortex-A53 is an Application Processing Unit (APU) to run OS and general purpose applications. The ZCU102 has a Zynq UltraScale+ XCZU9EG-2FFVB1156 MPSoC chip. The Cortex-A53 is an Arm v8 architecture-based 64-bit quad-core multiprocessing CPU. The Cortex-R5 is a Real-time Processing Unit (RPU) and based on an Arm v7 architecture 32-bit RPU with a dedicated tightly coupled memory (TCM). The Mali-400 is a graphics processing unit with pixel and geometry processor and 64 KB L2 cache.

Table 2.1: Xilinx Zynq UltraScale+ ZCU102 programmable logic resources.

Resource type	**Available**
BRAM	912
DSP	2,520
FF	548,160
LUT	274,080

The PL resources of the ZCU102 has been shown in Table 2.1. It has 912 block RAM (BRAM), 548160 flip-flops (FF), 2520 digital signal processing (DSP) units, and 274080 Look-up tables (LUT).

Vitis HLS (formerly Vivavo HLS) is a high-level synthesis tool that allows compilation of C, C++, and OpenCL functions to hardware modules using the device logic fabric and RAM/DSP blocks. Vitis HLS supports to develop a Register-transfer level (RTL) Intellectual Property (IP) for Xilinx devices in C/C++ programming languages.

Vitis HLS generate automatically the optimization in the C/C++ code. To achieve the best optimized RTL code that, it is necessary to add some directives to the code. The directives help the compiler to optimize design, reduce latency, I/O ports usage, and resource usages. Directives can be added in the Vivado HLS using the pragma keyword.

After synthesis is done, a report generated in the Vitis HLS. This report contains the generated RTL design modules, resource usages, and the computational cost.

If the synthesis was successful, then an IP block can be generated in the IP flow. After the IP is generated, an embedded system can be created in the Vivado IP integrator. That means the generated digital circuit connected to the other parts of the system via AXI4 buses such as memories, CPUs, other IPs, and peripheries. The bit stream file is generated based on the created embedded system in the Vivado IP integrator.

PYNQ is a platform which is running on the Xilinx ZCU102 FPGA board. After the Vivado IP integrator generated the circuit, this platform make it possible to call it as a function in Python language and run the implemented algorithms.

Vitis AI [16, 17] can accelerate AI inference on Xilinx hardware platforms such as FPGAs, SoCs, and Versal Adaptive Compute Acceleration Platforms (ACAP). The development environment includes of optimized IP cores, tools, libraries, models, and example designs. It makes possible to accelerate a Neural Network on FPGA without special FPGA knowledge.

The Tobii Pro Glasses 2 eye tracking system [18] has been used to record the videos and the gaze points during the experiments. The system has a lightweight Tobii Pro Glasses Head Unit, a wearable Tobii Pro Glasses Recording Unit and Tobii Glasses Controller (running on Windows 7, 8 or newer operating system) or Tobii Pro Glasses 2 API which is running on any devices. The Tobii Pro Glasses Recording unit can be worn as a glass.

The Tobii Pro Glasses Recording Unit can record an egocentric video 1920px $\times$ 1080px size with 25fps. It can also record the user's gaze point location and the distance between the glass and the looked object in millimeter.

In this research, Grasping in the Wild (GITW) [19] dataset has been used. This contains 404 short egocentric videos about a person, who want to grasp different objects namely, for example a bowl, or a coke can, or a pan in a kitchen.

Chapter 3

Contributions of the PhD research

In this chapter, I will present my main results and outline the perspectives of this work.

In this PhD research, I have developed a full solution for object recognition with Deep NNs in ego-centered video on a hybrid architecture using FPGA. The solution is guided by eye-tracker gaze fixations recordings and is designed for visual servoing of upper limb neuroprostethic arms. In the following, I summarize my contributions.

3.1 New scientific results

My scientific results are two-fold:

- 1. I have developed a hybrid solution: FPGA-CPU - for object recognition in ego-centered video by a Gaze-Driven CNN.
- 2. As a re-usable part of it I have implemented, on FPGA, a new SIFT detector for pre-processing of gaze data, namely their alignment in the current video frame.

I present my contributions in the following theses.

Thesis 1: A hybrid solution for Gaze-Driven CNN. [J1]

The backbone of the solution is ResNet50 [20]. A Reduction Layer has been introduced to accelerate computations. Faster R-CNN [21], is implemented for classification of object proposals, i.e. candidates bounding boxes in the current frame fitting the object of interest. Multiple Instance Learning (MIL) aggregation is applied for fusion of individual classification results to get the overall object score and position in the current frame. The recognition of an object type and object localization in a current video frame is running almost in real-time.

Thesis 1.1 Based on a critical analysis of computation complexity hybrid implementation of building blocks are proposed.[J1]

I have conducted a detailed computational time analysis of software implementation of object detection and localization method proposed by LaBRI, [22]. I have partitioned the system between the FPGA fabric and the ARM Cortex A53 processors of the Xilinx ZCU102 development board, based on the computing speed measurements of the building blocks. As a reference, the computing time of each image processing step was measured on a laptop microprocessor and its power dissipation was estimated.

The gaze point alignment is fast enough according to my measurements on the ARM Cortex A53 [23]-embedded CPU, except the SIFT [1] point extraction step. Therefore, I have implemented the SIFT detection module on the programmable logic part of the Xilinx ZCU102 [15] FPGA board

The gaze-driven CNN is built on four different modules: ResNet50 [20], Reduction Layer, Faster R-CNN [21], and Multiple Instance Learning (MIL) aggregation. I accelerated the ResNet50 [20] on FPGA with Vitis AI because the measured computational speed on the ARM Cortex A53 processor was only 0.55 fps. I have improved it to 37.23 fps due to this implementation.

Thesis 1.2: I proposed a new Reduction Layer between ResNet50 and Faster R-CNN in the original algorithm to reduce the number of input channels for the Faster R-CNN block. [J1]

The frame rate can be increased to 25 fps when the number of input channels for the Faster R-CNN is reduced to 128 by the Reduction Layer. The experiments show that the accuracy using only 128 channels is still high enough for the bounding box classification.

Thesis 1.3: My hybrid FPGA + ARM solution show that a wearable device is achievable with low power dissipation and with real-time processing speed. [J1]

My experimental setup, with the whole chain of modules, ensures the computational speed of 117.175 ms on average per video frame. That is, the computational frame rate is approximately 8.5 fps. It is yet not suitable for real-time processing, as the required video frame-rate for servoing of a prosthetic arm is 10 fps. However, this computing time can be improved by pipelining the system. Because each block of the system can finish processing of a video frame within 40 ms, according to our measurements. This means that we can achieve a frame rate of 25 fps. However, the latency of the system will be increased to 117.175 ms, which is still affordable in this scenario.

Thesis 2: I have developed a hybrid solution with a 32 bit floating point computations for SIFT keypoint extractor. It runs on FPGA and generates the same results as the OpenSIFT software implementation. [J2]

In the overall system SIFT [1] point detection is used for the alignment of gaze points. It is a critical block according to our computation time measurements, because the processing time of a frame was 72.407 ± 3.349 ms on Intel i5 7300HQ CPU. Therefore, we have designed, optimized and implemented it on FPGA.

Thesis 2.1: My proposed FPGA solution for the SIFT point detector can be implemented on Xilinx ZCU102 FPGA-board. [J2]

I have implemented a FPGA-optimizied SIFT keypoint detector on Xilinx ZCU102. The designed digital circuit has been used 117,620 of the 274,080 available LUTs resources, 157,946 of the 548,160 available FFs resources, 416.5 of the 912 available BRAMs resources and 938 of the 2,520 available DSPs resources. There are enough free resources left on the Xilinx ZCU102 to develop more computer vision algorithms on FPGA.

Thesis 2.2: In the proposed FPGA SIFT implementation, I have made a simplification in the keypoint localization step. The FPGA optimzed SIFT which gives close results to the reference SIFT point detector. [J2]

Instead of computing Taylor expansion for precise SIFT point localization, keypoints too close to each other are filtered using Non-maximum Supression. This approach is not changing the accuracy of the original algorithm accordingly to our comparison. I have compared my hardware/software solution to other hardware or hybrid implementations of the SIFT algorithm and with the baseline software detector OpenSIFT. I have conducted computational experiments on a large set of 3860 video frames to validate the implemented detector. My algorithm implemented on FPGA is giving an average precision of 0.84 and the average recall of 0.94 in SIFT-point detection compared to the baseline OpenSIFT.

Thesis 2.3: I have experimentally shown that the proposed FPGA SIFT implementation is time-efficient and the power consumption is low. The processing rate of the implemented SIFT detector is 135 images per second, when the input image resolution is of 480px × 480px, and it is running on Xilinx ZCU102 FPGA board. The total power consumption of my FPGA SIFT implementation is 5.6W, which is suitable for wearable devices. [J2]

I used these results in Thesis 1, because in one of the steps, namely in gaze point alignment the SIFT keypoint extraction is a crucial step.

3.2 Perspectives

The system integration is possible, because the obtained accuracy is high enough for real world demonstration. Currently, the visual block is accelerated on FPGA, but it is necessary to implement the system servoing steps on an embedded hybrid system.

The algorithm can also be developed further. The current algorithm is extracting objects in video frame by frame. However, it is possible to achieve higher accuracy with tracking. In that case, faster processing time is also achievable. Another possible future work will be to use move-to-data incremental learning method [24]. With this method, it is possible to adapt this system for a changing living environment of a person. It will also be adaptable for a different environment.

A wearable device is required in that case to control a prosthetic arm. To do that, a technology needed has to have a low power consumption and a high computational speed. It is feasible, as my experiments have shown.

It is possible to increase the computational speed with pipelining. Pipelining is a method when a step finishes, for example gaze-point alignment of a frame, then it is possible to start gaze-point alignment on the next frame and the gaze-point noise reduction can be done on the original frame at the same time. However, this would increase the latency of the system. The current system latency is around 117.175 ± 6.8 ms, which is the accepted latency allowed by the control of the robotic arm ($\sim$100 ms).

List of publications

Journal publications:

[J1] Attila Fejér, Zoltán Nagy, Jenny Benois-Pineau, Péter Szolgay, Aymar de Rugy, and Jean-Philippe Domenger. Hybrid fpga-cpu-based architecture for object recognition in visual servoing of arm prosthesis. *Journal of Imaging*, 8(2), 2022.

[J2] A. Fejér, Z. Nagy, J. Benois-Pineau, P. Szolgay, A. de Rugy, and J-P. Domenger. Implementation of scale invariant feature transform detector on fpga for low-power wearable devices for prostheses control. *Int J Circ Theor Appl*, 49:2255 – 2273, 2021.

Conference publications:

[C1] Attila Fejér, Zoltán Nagy, Jenny Benois-Pineau, Péter Szolgay, Aymar de Rugy, and Jean-Philippe Domenger. Fpga-based sift implementation for wearable computing. In *2019 IEEE 22nd International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*, pages 1–4, 2019.

[C2] Attila Fejér, Zoltán Nagy, Jenny Benois-Pineau, Péter Szolgay, Aymar de Rugy, and Jean-Philippe Domenger. Array computing based system for visual servoing of neuroprosthesis of upper limbs. In *2021 17th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*, pages 1–5, 2021.

Other publications:

[Au1] Attila Fejér, Zoltán Nagy, Jenny Benois-Pineau, Péter Szolgay, Aymar de Rugy, and Jean-Philippe Domenger. Fpga-based sift implementation. In *2. Francia-Magyar Tudományos Kutatói Fórum*, 2019.

[Au2] Attila Fejér, Zoltán Nagy, Jenny Benois-Pineau, Péter Szolgay, Aymar de Rugy, and Jean-Philippe Domenger. Array computing based system for visual servoing of neuroprosthesis of upper limbs. In *2019 Research in hybrid control of neuro-prosthetics on French-Swiss-Hungarian workshop*, pages 19–20, 2019.

[Au3] Attila Fejér. Image processing algorithms implementation on fpga. In *PhD Proceedings Annual Issues of the Doctoral School, Faculty of Information Technology and Bionics, Pázmány Péter Catholic University*, page 33, 2019.

[Au4] Attila Fejér. Evaluation of fpga based sift implementation by using real-life examples. In *PhD Proceedings Annual Issues of the Doctoral School, Faculty of Information Technology and Bionics, Pázmány Péter Catholic University*, page 43, 2020.

[Au5] Attila Fejér. Heterogenous architecture for visual servoing a prosthetic arm. In *PhD Proceedings Annual Issues of the Doctoral School, Faculty of Information Technology and Bionics, Pázmány Péter Catholic University*, page 44, 2021.

[Au6] Attila Fejér. Recognition object to grasp by using a hybrid system. In *PhD Proceedings Annual Issues of the Doctoral School, Faculty of Information Technology and Bionics, Pázmány Péter Catholic University*, page 44, 2022.

References

[1] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov 2004. 1, 3.1

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 1

[3] Rob Hess. An open-source siftlibrary. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, page 1493–1496, New York, NY, USA, 2010. Association for Computing Machinery. 1

[4] Z. Li, H. Jia, and Y. Zhang. Hartsift: A high-accuracy and real-time sift based on gpu. 2017 IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS), pages 135–142, 2017. 1

[5] Mårten Björkman, Niklas Bergström, and Danica Kragic. Detecting, segmenting and tracking unknown objects using multi-label mrf inference. *Computer Vision and Image Understanding*, 118:111 – 127, 2014. 1

[6] Ginés Doménech-Asensi, Juan Zapata-Pérez, Ramón Ruiz-Merino, José Alejandro López-Alcantud, José Ángel Díaz-Madrid, Víctor Manuel Brea, and Paula López. All-hardware sift implementation for real-time vga images feature extraction. *Journal of Real-Time Image Processing*, 17(2):371–382, Apr 2020. 1

[7] Pablo Rubio-Ibáñez, Ramón Ruiz-Merino, Ginés Doménech-Asensi, José Javier Martínez-Álvarez, Juan Zapata-Pérez, José Ángel Díaz-Madrid, and José Alejandro López-Alcantud. An all-hardware implementation of the

subpixel refinement stage in sift algorithm. *International Journal of Circuit Theory and Applications*, 46(9):1690–1702, 2018. 1

[8] John Vourvoulakis, John Kalomiros, and John Lygouras. Fpga accelerator for real-time sift matching with ransac support. *Microprocessors and Microsystems*, 49:105 – 116, 2017. 1

[9] John Vourvoulakis, John Kalomiros, and John Lygouras. Fully pipelined fpga-based architecture for real-time sift extraction. *Microprocessors and Microsystems*, 40:53–73, 2016. 1

[10] Leonardo Chang, José Hernández-Palancar, L. Enrique Sucar, and Miguel Arias-Estrada. Fpga-based detection of sift interest keypoints. *Machine Vision and Applications*, 24(2):371–392, Feb 2013. 1

[11] A-jun Shao, Qian Wei-xian, Gu Guo-hua, and Lu Kai-li. Real-time implementation of SIFT feature extraction algorithms in FPGA. volume Proc. SPIE 9622 of *2015 International Conference on Optical Instruments and Technology: Optoelectronic Imaging and Processing Technology*, pages 233 – 247. International Society for Optics and Photonics, SPIE, 2015. 1

[12] Angel Rodríguez-Vázquez, Rafael Domínguez-Castro, Francisco Jiménez-Garrido, Sergio Morillas, Alberto García, Cayetana Utrera, Ma. Dolores Pardo, Juan Listan, and Rafael Romay. A CMOS vision system on-chip with multi-core, cellular sensory-processing front-end. Cellular Nanoscale Sensory Wave Computing, pages 129–146. Springer US, oct 2009. 1

[13] M. Suárez, V. M. Brea, J. Fernández-Berni, R. Carmona-Galán, D. Cabello, and A. Rodríguez-Vázquez. Gaussian pyramid extraction with a cmos vision sensor. 2014 14th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA), pages 1–2, July 2014. 1

[14] Toshiba. Sps. http://www.toshiba-teli.co.jp/en/products/industrial/sps/sps.htm, 2019. 1

[15] Xilinx. Ug1182 zcu102 evaluation board - user guide, 6 2019. 2, 3.1

[16] Vinod Kathail. Xilinx vitis unified software platform. In *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '20, page 173–174, New York, NY, USA, 2020. Association for Computing Machinery. 2

[17] Xilinx. Ug1414 (v2.5): Vitis ai user guide, 6 2022. 2

[18] Tobii glass. `https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/`. Accessed: 2018-05-05. 2

[19] LaBRI. Grasping in the wild, 2016. 2

[20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 3.1

[21] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017. 3.1

[22] Iván González-Díaz, Jenny Benois-Pineau, Jean-Philippe Domenger, Daniel Cattaert, and Aymar de Rugy. Perceptually-guided deep neural networks for ego-action prediction: Object grasping. *Pattern Recognition*, 88:223–235, 2019. 3.1

[23] Xilinx. Ds891 - zynq ultrascale+ mpsoc data sheet: Overview, 5 2021. 3.1

[24] Miltiadis Poursanidis, Jenny Benois-Pineau, Akka Zemmari, Boris Mansenca, and Aymar de Rugy. Move-to-data: A new continual learning approach with deep cnns, application for image-class recognition, 2020. 3.2