A megfogandó tárgy kiválasztása és felismerése robotkar vezérléséhez

Fejér Attila

PhD Tézisfüzet

Témavezető:
Dr. Szolgay Péter
Konzulens:
Dr. Benois-Pineau, Jenny

Roska Tamás Műszaki és Természettudományi Doktori Iskola
Pázmány Péter Katolikus Egyetem
Doctoral School of Mathematics and Computer Science
University of Bordeaux

Budapest, 2022

Tartalomjegyzék

1. fejezet

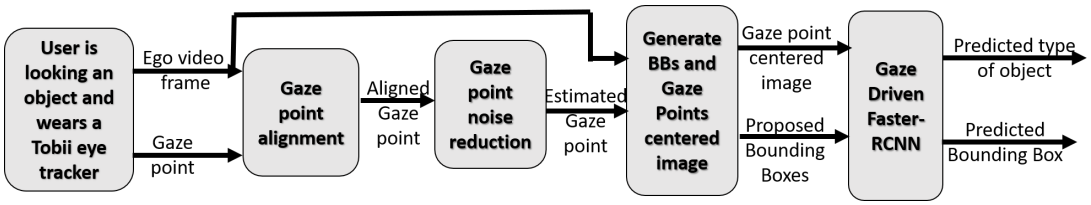
Bevezetés és nyitott kérdések

A karprotézisek használata napjainkban elérhető és hasznos eszközzé vált azok számára, akik baleset vagy különböző betegségek miatt elvesztették felső végtagjaikat. A technológia egyre alkalmazkodóbb és jobb berendezések kifejlesztéséhez vezet, amelyek segítik a felhasználó mindennapi életét. A mi elképzelésünk az, hogy tovább javítsuk a karprotézisek minőségét a vizuális információk felhasználásával. Vizuális vezérlő mechanizmusunknak alkalmasnak kell lennie az eszköz viselésre, és lehetővé kell tennie a valós idejű feldolgozást, miközben nem korlátozza a művégtag és annak viselőjének mobilitását. Ezeket a követelményeket alaposan megvizsgálva úgy döntöttünk, hogy a vezérlő mechanizmust FPGA-ra fejlesztjük ki. Ez azonban egy nyitott probléma, és intenzív kutatási tevékenység folyik ezen a területen.

A vezérlő program több önálló részből áll, amelyek egy számítógépen futnak. Célunk az algoritmusok felgyorsítása a valós idejű feldolgozási sebesség elérése érdekében.

Ez egy összetett számítógépes látás, képfeldolgozás és robotikai probléma, hiszen meg kell találnunk azt a tárgyat, amit a felhasználó meg akar fogni. Ez a rész egy számítógépes látás és képfeldolgozási feladat. A robotkart is szeretnénk irányítani, hogy segítsen a felhasználónak megragadni ezt a tárgyat.

Az amputáltak protéziskarjának vezérlésére egy hibrid hardver-szoftver megoldást javasoltunk. A rendszer az 1.1 ábrán látható, amely a fő komponenseket mutatja be: A képkockák és a tekintetpontok rögzítése Tobii eye-trackerrel, a tekintetpontok összehangolása, a tekintetpontok zajcsökkentése, a zajcsökkentett



1.1. ábra. A műkar vizuálisan irányított rendszere.

tekintetpontok körüli befoglaló téglalapok létrehozása, a tárgyak észlelése a Tekintetvezérelt (Gaze-Driven) CNN segítségével.

Tobii eye-tracker, amely rögzíti a felhasználó tekintetpontjait (azt a pontot, ahová a felhasználó éppen néz). A tekintetpontok 3 koordinátával rendelkeznek, x,y,gamma. Ahol a gamma a tekintetpont mélysége. A Tobii eye-tracker egy egocentrikus nézeti videót is rögzít. A rögzített videó felbontása 1920 px x 1080 px (Full HD), a képkockasebesség pedig 25 fps.

A következő lépés a tekintetpontok összehangolása a videó minden egyes egymást követő képkockájának Scale-Invariant Feature Transform (SIFT) kulcspontjainak kinyeréséhez. Méréseink megmutatták, hogy a SIFT [1] volt az egyik leglassabb rész az algoritmusaink folyamában.

A SIFT egy széles körben használt és alkalmazott kulcspontdetektor. Létezik CPU[2, 3], GPU[4, 5] és FPGA[6, 7, 8, 9, 10, 11] implementációja is. Az OpenCV SIFT könyvtár[2] és az OpenSIFT[3] népszerű keretrendszerek a SIFT kulcspontok kinyerésére és a leírók számítására CPU-n. E CPU-s implementációk futási ideje azonban túl lassú egy beágyazott rendszeren történő valós idejű képfeldolgozáshoz. A számítás felgyorsítható például GPU-k használatával: A CudaSIFT[5] egy 1280px-es × 960px-es képet 12,7 ms, azaz 78,74 képkocka/másodperc (fps) alatt képes feldolgozni az NVIDIA GeForce GTX 580 GPU-n. Az energiafogyasztása azonban 244 W, ami túl magas egy viselhető alkalmazáshoz. A HartSIFT[4] az NVIDIA GeForce GTX TITAN Black processzoron a bemeneti kép méretétől függően 3,14∼10,57 ms (94,61-318 fps) alatt képes kivonni a jellemzőket. Ennek a GPU-nak az energiafogyasztása 250 W, amely nem megfelelő egy hordozható eszköz létrehozásához.

A SIFT kulcspont meghatározás számításigényesebb művelete a Gauss-piramisok kiszámítása, mivel ehhez a Gauss-szűrők együtthatóinak a skálatérbeli képek-

kel való szorzására van szükség. Erre a lépésre Rodríguez-Vázquez [12, 13] egy analóg megoldást dolgozott ki, ahol a Gauss-piramis számítása analóg CMOS-áramkörrel történik, méghozzá nagyon alacsony disszipált teljesítmény mellett. Ez a párhuzamos feldolgozás nagy számítási teljesítményt eredményez az analóg VLSI megvalósításoknál. A tömb mérete azonban kicsi [12, 14, 13]. A [13] analóg érzékelő/processzor megvalósítás 88x60 feldolgozóelemmel rendelkezik, és minden egyes feldolgozóelem 4 fotodiódával rendelkezik. A számítási egység a kamera látásérzékelő egységéhez van csatlakoztatva. A látásérzékelő tömb csak 176x120 pixeles, és a rendszer 0,18 μm CMOS technológiával valósul meg. Ez a megoldás megfelelő feldolgozási teljesítménnyel rendelkezik, de az analóg érzékelő felbontása nem elég nagy a mi alkalmazásunkhoz, mivel a módszerünk során 480px $\times$ 480px-es képre van szüksége bemenetként.

Számos különböző SIFT implementációt publikáltak FPGA-n, mint például a Doménech-Asensi által tervezett rendszer [6], amely az algoritmus egyszerűsített változata. Feltételezik, hogy minden egyes jellemzőpontnak legfeljebb két fő orientációja van, az orientációs hisztogram 36 helyett 8 bint használ. Így a bonyolultság csökken, de az orientációk pontossága kisebb az eredeti SIFT[1]-hez képest. A rendszer a 640px $\times$ 480px méretű bemeneti képeket 99 fps feldolgozási sebességgel dolgozza fel a Xilinx Virtex-5 rendszeren, fixpontos ábrázolást használva.

A SIFT kulcspontok birtokában a következő lépés a FLANN illesztés. Van egy referenciakép, amelyet összehasonlítunk a referenciakép-10, referenciakép- ...., referenciakép-1 képekkel, és ugyanabba a síkba helyezzük őket. Ezután kiszámíthatjuk a homográfia mátrixot, és ezzel megbecsülhetjük a tekintetpontok helyét a referenciakép síkjában.

A tekintetpontok zajcsökkentése során a cél a kiugró tekintetpontok kiszűrése. A kiugró tekintetpontokat a szakkádok és a fejmozgások okozzák, amikor a felhasználó megpróbálja megtalálni a tárgyat, vagy amikor a megragadás folyamata során elvonják a figyelmét.

A fixált tekintetpont körül befoglaló téglalapokat generálunk. Kilenc különböző méretarányú és méretű befoglaló téglalap generálódik, ahol a középpont a fixált tekintetpont. A tekintet alapú CNN megbecsüli a tárgy típusát és a tárgy

helyét. E CNN bemenetei a 9 befoglaló téglalap és az aktuális képkocka. A tárgy típusára és helyére a protéziskar jobb irányításához van szükség.

2. fejezet

Kutatási módszertan

Célunk egy olyan eszköz létrehozása, amely képes egy robotkart vezérelni kis súllyal és alacsony energiafogyasztással. Másrészt hatékonyan kell működnie, hogy az adatokat gyorsan feldolgozza.

Az FPGA-t (Field Programmable Gate Arrays) azért választottuk, mert alacsony az energiafogyasztása, és ezért alkalmas egy hordható eszköz prototípusának elkészítésére. Ebben az esetben a felhasználó az alkalmazásunkat viselni fogja, ezért az eszköznek könnyűnek kell lennie. Az FPGA rendelkezik a szükséges számítási teljesítménnyel a felhasznált algoritmusok felgyorsításához, hogy az valós időben történjen.

A Xilinx Zynq UltraScale+ MPSoC ZCU102 [15] fejlesztőkártya lett kiválasztva a prototípus lefejlesztéséhez.

A ZCU102 lapon található XCZU9EG FPGA eszköz egy feldolgozórendszer (PS) és egy programozható logikai (PL) részből áll. A PS rész négymagos Arm Cortex-A53, kétmagos Cortex-R5F valós idejű processzorokkal és egy Mali-400 MP2 grafikus feldolgozóegységgel rendelkezik. A Cortex-A53 egy alkalmazásfeldolgozó egység (APU) az operációs rendszer és az általános célú alkalmazások futtatására. A ZCU102 egy Zynq UltraScale+ XCZU9EG-2FFVB1156 MPSoC chipet tartalmaz. A Cortex-A53 egy Arm v8 architektúrán alapuló 64 bites négymagos, többprocesszoros CPU. A Cortex-R5 egy valós idejű feldolgozó egység (RPU), és egy Arm v7 architektúrájú 32 bites RPU-n alapul, dedikált szorosan kapcsolt memóriával (TCM). A Mali-400 egy grafikus feldolgozó egység pixel- és geometriaprocesszorral és 64 KB L2 gyorsítótárral.

2.1. táblázat. Xilinx Zynq UltraScale+ ZCU102 programozható logikai erőforrások.

Erőforrás típusa	**Elérhető erőforrás**
BRAM	912
DSP	2 520
FF	548 160
LUT	274 080

A ZCU102 PL-erőforrásai a 2.1 táblázatban szerepelnek. Ez 912 blokk RAM (BRAM), 548160 flip-flop (FF), 2520 digitális jelfeldolgozó (DSP) egység és 274080 Look-up tábla (LUT).

A Vitis HLS (korábban Vivado HLS) egy magas szintű szintéziseszköz, amely lehetővé teszi a C, C++ és OpenCL funkciók hardvermodulokká történő fordítását az eszköz programozható logikai és a RAM/DSP blokkokat használva. A Vitis HLS támogatja a Register-transfer level (RTL) Intellectual Property (IP) fejlesztését Xilinx eszközökhöz C/C++ programozási nyelveken.

A Vitis HLS automatikusan generálja aa megoldást a C/C++ kódban. A legjobb optimalizált RTL kód eléréséhez szükséges néhány direktívát is hozzáadni a kódhoz. A direktívák segítenek a fordítónak a tervezés optimalizálásában, a késleltetés, az I/O portok használatának és az erőforrás-felhasználásnak a csökkentésében. A direktívákat a Vivado HLS-ben a pragma kulcsszóval lehet hozzáadni.

A szintézis elvégzése után egy összegzés készül a Vivado HLS-ben. Ez tartalmazza a generált RTL tervezési modulokat, az erőforrás-felhasználásokat és a számítási költségeket.

Ha a szintézis sikeres volt, akkor egy IP blokk generálható az IP flow-ban. Az IP generálása után a Vivado IP-integrátorban létrehozható egy beágyazott rendszer. Ez azt jelenti, hogy a generált digitális áramkör AXI4 buszokon keresztül kapcsolódik a rendszer többi részéhez, például memóriákhoz, CPU-khoz, más IP-khez és perifériákhoz. A bitfolyam fájl a létrehozott beágyazott rendszer alapján a Vivado IP integrátorban generálódik.

A PYNQ egy olyan platform, amely a Xilinx ZCU102 FPGA lapkán fut. Miután a Vivado IP-integrátor létrehozta az áramkört, ez a platform lehetővé

teszi, hogy Python nyelven függvényként hívja meg, és futtassa a megvalósított algoritmusokat.

A Vitis AI [16, 17] képes felgyorsítani az Mesterséges Intelligencia (MI) algoritmusokat a Xilinx hardverplatformjain, például az FPGA-kon, SoC-ken és a Versal Adaptive Compute Acceleration Platformokon (ACAP). A fejlesztőkörnyezet optimalizált IP-magokat, eszközöket, könyvtárakat, modelleket és példaterveket tartalmaz. Lehetővé teszi a neurális hálózat FPGA-n történő gyorsítását speciális FPGA-ismeretek nélkül.

A kísérletek során a Tobii Pro Glasses 2 eye-tracker rendszert [18] használtak a videók és a tekintetpontok rögzítésére. A rendszer egy könnyű Tobii Pro Glasses fejegységgel, egy viselhető Tobii Pro Glasses videó felvételt rögzítő egységgel és Tobii Glasses vezérlővel (Windows 7, 8 vagy újabb operációs rendszeren fut) vagy Tobii Pro Glasses 2 API-val rendelkezik, amely bármilyen eszközön fut. A Tobii Pro Glasses Recording Unit szemüvegként viselhető.

A Tobii Pro Glasses Recording Unit képes Full HD (1920px × 1080px) méretű egocentrikus videó rögzítésére 25fps sebességgel. Emellett rögzíti a felhasználó tekintetpontjának helyét és a szemüveg és a nézett tárgy közötti távolságot milliméterben.

Ebben a kutatásban a Grasping in the Wild (GITW) [19] adathalmazt használtuk. Ez 404 rövid egocentrikus videót tartalmaz egy személyről, aki különböző tárgyakat akar megfogni, például egy tálat, vagy egy kólásdobozt, vagy egy serpenyőt a konyhában.

3. fejezet

A doktori kutatás hozzájárulásai

Ebben a fejezetben bemutatom a főbb eredményeimet, és felvázolom a munka perspektíváit.

Ebben a doktori kutatásban teljes megoldást fejlesztettem ki a Deep NN-ekkel történő objektumfelismerésre egocentrikus videókban egy FPGA-t használó hibrid architektúrán.

3.1. Új tudományos eredmények

Tudományos eredményeim két részre osztottam:

- Kidolgoztam egy hibrid megoldást: FPGA-CPU - egocentrikus videóban lévő tárgyak felismerésére a Tekintetvezérelt (Gaze-Driven) CNN segítségével.

- Ennek újrafelhasználható részeként FPGA-n implementáltam egy új SIFT-detektort a tekintetadatok előfeldolgozására, nevezetesen a tekintet helyének a pontos meghatározására az aktuális videóképben.

Hozzájárulásaimat a következő tézisekben mutatom be.

1. tézis: Egy hibrid megoldású Tekintetvezérelt CNN-t terveztem [J1]

A Tekintetvezérelt CNN több elemből áll, ennek az első része a ResNet50 [20] amely a bemeneti képekről kinyeri a képre jellemző tulajdonságokat. A következő

modulja a Redukciós Réteg (Reduction Layer), amelyet a ResNet50 és a Faster R-CNN között vezettünk be, hogy a számítási igényt csökkentse. A Faster R-CNN [21], az objektumjavaslatok, azaz az aktuális képkockában lévő, az érdeklődésre számot tartó objektumra illeszkedő jelölt határolókeretek osztályozására szolgál. Az egyes osztályozási eredmények fúziójára a többpéldányos tanulás (Multiple Instance Learning (MIL)) aggregációt alkalmazunk, hogy megkapjuk az objektum pontszámát és az aktuális képkockában elfoglalt pozícióját. A tárgytípus felismerése (serpenyő, tál stb.) és a tárgy lokalizálása az aktuális videóképben kvázi valós időben történik.

1.1 tézis: A számítási komplexitás elemzése alapján az építőelemek hibrid megvalósítását javasoltam.[J1]

Részletes számítási időelemzést végeztem a LaBRI [22] által javasolt objektumdetektálási és lokalizációs rendszer szoftveres implementációjáról. Az alkalmazást az építőblokkok számítási sebességének mérései alapján felosztottam az FPGA programozható logika és a Xilinx ZCU102 fejlesztői kártya ARM Cortex A53 processzorai között. Referenciaként az egyes képfeldolgozási lépések számítási idejét egy laptop mikroprocesszorán mértem, és megbecsültem az energiafelhasználást.

A Tekintetvezérelt optimalizált algoritmusa méréseim szerint elég gyors az ARM Cortex A53 [23] CPU-n, kivéve a SIFT [1] kulcspont detektálás. Ezért a SIFT kulcspont detektáló modult a Xilinx ZCU102 [15] FPGA programozható logikai részébe implementáltam.

A Tekintetvezérelt CNN négy különböző modulra épül: ResNet50 [20], Redukciós Réteg (Reduction Layer), Faster R-CNN [21] és MIL aggregáció. Felgyorsítottam a ResNet50 [20] FPGA-n a Vitis AI segítségével, mivel az ARM Cortex A53 processzoron mért számítási sebesség mindössze 0,55 fps volt. Ennek az implementációnak köszönhetően 37,23 fps-re gyorsult.

1.2. tézis: Az eredeti algoritmusban a ResNet50 és a Faster R-CNN közé egy új Redukciós Réteget (Reduction Layer) javasoltam a Faster R-CNN blokk bemeneti csatornáinak csökkentése érdekében.[J1]

A képkockasebesség 25 fps-re növelhető, ha a Faster R-CNN bemeneti csatornáinak számát a Redukciós Réteg 128-ra csökkenti. A kísérletek azt mutatják,

hogy a 128 csatornát használó pontosság még mindig elég magas a befoglaló téglalap alapú klasszifikálásához.

1.3. tézis: A hibrid FPGA + ARM megoldásom azt mutatja, hogy egy mobil eszköz alacsony energiafogyasztással és valós idejű feldolgozási sebességgel megvalósítható.[J1]

Kísérleti rendszerem az összes modulra mért átlagosan 117,175 ms számítási sebességet biztosít videóképenként. Ez azt jelenti, hogy a számítási képkockasebesség körülbelül 8,5 fps. Ez még nem alkalmas valós idejű feldolgozásra, mivel a protéziskar szervózásához szükséges videóképkocka sebesség 10 fps. Ez a számítási idő azonban javítható a rendszer pipelinosítása révén. Ugyanis a rendszer egyes blokkjai méréseink szerint 40 ms alatt képesek befejezni egy videóképkocka feldolgozását. Ez azt jelenti, hogy 25 fps képkockasebességet érhetünk el. A rendszer késleltetése azonban 117,175 ms-ra nő, ami megfelelő.

2. tézis: Kidolgoztam egy hibrid megoldást 32 bites lebegőpontos számításokkal a SIFT algoritmushoz. Ez FPGA-n fut, és ugyanazokat az eredményeket produkálja, mint az OpenSIFT szoftveres implementáció. [J2]

A teljes rendszerben a SIFT [1] pontdetektálást használjuk a tekintetpontok összehangolására. Számítási időméréseink szerint ez egy kritikus blokk, mivel egy képkocka feldolgozási ideje 72,407 ± 3,349 ms volt Intel i5 7300HQ CPU-n. Ezért terveztük, optimalizáltuk és FPGA-n valósítottuk meg.

2.1. tézis: A SIFT kulcspontdetektorhoz Xilinx ZCU102 FPGA-n megvalósítható megoldást javasoltam. [J2]

Egy FPGA-optimalizált SIFT kulcspontdetektort valósítottam meg Xilinx ZCU102-n. A tervezett digitális áramkör a rendelkezésre álló 274 080 LUTs erőforrásból 117 620, az 548 160 rendelkezésre álló FF erőforrásokból 157 946, a 912 rendelkezésre álló BRAM erőforrásokból 416,5 és a 2520 rendelkezésre álló DSPs erőforrásokból 938 került felhasználásra. A Xilinx ZCU102-n még elegendő szabad erőforrás maradt ahhoz, hogy további számítógépes látás algoritmusokat fejlesszünk az FPGA-ra.

2.2. tézis: A javasolt FPGA SIFT implementációban egyszerűsítettem a kulcspont lokalizációs lépést. Az FPGA optimalizált SIFT,

amely a referencia SIFT kulcspontdetektorhoz közeli eredményeket ad. [J2]

A pontos SIFT-kulcspontlokalizációhoz szükséges Taylor-sorfejtés kiszámítása helyett az egymáshoz túl közel eső kulcspontokat a Nem-maximumok elhagyásának segítségével szűrjük. Ez a megközelítés az összehasonlításunknak megfelelően nem változtatja meg az eredeti algoritmus pontosságát. A hardveres/szoftveres megoldásomat összehasonlítottam a SIFT-algoritmus más hardveres vagy hibrid implementációival és az OpenSIFT alapszintű szoftveres detektorral. Számítási kísérleteket végeztem egy nagy, 3860 videóképből álló halmazon, hogy validáljam a megvalósított detektort. Az FPGA-n implementált algoritmusom átlagosan 0,84-es pontosságot és 0,94-es átlagos visszahívást eredményez a SIFT-pontok detektálásában az OpenSIFT referencia szoftverhez képest.

2.3. tézis: Kísérletileg kimutattam, hogy a javasolt FPGA SIFT implementáció időhatékony és alacsony az energiafogyasztása. A megvalósított SIFT-detektor feldolgozási sebessége 135 kép másodpercenként, ha a bemeneti kép felbontása 480px × 480px, és a Xilinx ZCU102 FPGA-n fut. Az FPGA SIFT implementációm teljes energiafogyasztása 5,6W, ami alkalmas viselhető eszközökhöz. [J2]

Ezeket az eredményeket használtam az 1. tézisben, mert az egyik lépésben, nevezetesen a tekintetpont-kiigazításban a SIFT kulcspontok kinyerése kulcsfontosságú lépés.

3.2. Perspektívák

Jelen rendszer integrálása lehetséges, mivel a kapott pontosság elég nagy a valós világbeli alkalmazásokhoz. Jelenleg a vizuális blokkot FPGA-n gyorsítják fel, de a rendszer vezérlési lépéseit beágyazott hibrid rendszeren kell megvalósítani.

Az algoritmus is továbbfejleszthető. A jelenlegi algoritmus képkockánként vonja ki az objektumokat a videóból. A követéssel azonban nagyobb pontosság érhető el. Ebben az esetben gyorsabb feldolgozási idő is elérhető. Egy másik lehetséges jövőbeli munka a Move-to-Data inkrementális tanulási módszer [24] alkalmazása lesz. Ezzel a módszerrel a rendszer adaptálható a személy változó lakókörnyezetéhez. A rendszer más környezethez is könnyen átalakítható lesz.

Mobil eszközre van szükség a protéziskar vezérléséhez. Ehhez a szükséges technológiának alacsony energiafogyasztással és nagy számítási sebességgel kell rendelkeznie. Ez megvalósítható, amint azt a kísérleteim megmutatták.

A számítási sebességet pipelininggel lehet növelni. A pipelining egy olyan módszer, amikor egy lépés befejeződik, például egy képkocka tekintetpont-kiigazítása, akkor a következő képkockán el lehet kezdeni a tekintetpont-kiigazítást, és a tekintetpont-zajcsökkentés az eredeti képkockán is elvégezhető egyszerre. Ez azonban növelné a rendszer késleltetési idejét. A rendszer jelenlegi késleltetési ideje körülbelül 117,175 $\pm$ 6,8 ms, ami a robotkar vezérlése által megengedett elfogadott késleltetés ($\sim$100 ms).

Publikációk listája

Folyóiratban megjelent publikációk:

[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.

Konferencia kiadványokban megjelent publikációk:

[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.

További publikációk:

[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.

Hivatkozások

[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