Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Nik Prinčič

# Vizualno sledenje na vgrajenih napravah

diplomsko delo

visokošolski strokovni študijski program
prve stopnje
računalništvo in informatika

Mentor: doc. dr. Luka Čehovin Zajc

Ljubljana, 2023



*Besedilo je oblikovano z urejevalnikom besedil LaTeX.*

**Kandidat:** Nik Prinčič
**Naslov:** Vizualno sledenje na vgrajenih napravah
**Vrsta naloge:** Diplomska naloga na visokošolskem programu prve stopnje Računalništvo in informatika
**Mentor:** doc. dr. Luka Čehovin Zajc

**Opis:**
Besedilo teme diplomskega dela študent prepiše iz študijskega informacijskega sistema, kamor ga je vnesel mentor. V nekaj stavkih bo opisal, kaj pričakuje od kandidatovega diplomskega dela. Kaj so cilji, kakšne metode naj uporabi, morda bo zapisal tudi ključno literaturo.

**Title:** Visual tracking on embedded devices

**Description:**
opis diplome v angleščini

*Na tem mestu zapišite, komu se zahvaljujete za pomoč pri izdelavi diplomske naloge oziroma pri vašem študiju nasploh. Pazite, da ne boste koga pozabili. Utegnil vam bo zameriti. Temu se da izogniti tako, da celotno zahvalo izpustite.*

Svoji dragi Alenčici.

# Kazalo

# Seznam uporabljenih kratic

| kratica | angleško | slovensko |
|---|---|---|
| **AI** | Artificial inteligence | Umetna inteligenca |
| **ANN** | Artificial neural network | Umetna nevronska mreža |
| **BNN** | Biological neural network | Biološka nevronska mreža |
| **DNN** | Deep Neural Network | Globoka nevronska mreža |
| **CNN** | Convolutional neural network | Konvolucijska nevronska mreža |
| **FPS** | Frames per second | Sličice na sekundo |
| **OAK** | OpenCV AI Kit | OpenCV AI komplet |
| **OAK** | OpenCV AI Kit | OpenCV AI komplet |
| **BBOX** | Bounding box | Omejitveni okvir |
| **IoT** | Internet of things | Internet stvari |
| **SOT** | Single object tracking | Sledenje posameznega objekta |
| **MOT** | Multiple object tracking | Sledenje več objektom |
| **VOT** | Visual object tracking | Vizualno sledenje |
| **VPU** | Visual processing unit | Vizualna procesna enota |
| **ROI** | Region of interest | Območje interesa |

# Povzetek

**Naslov:** Vizualno sledenje na vgrajenih napravah

**Avtor:** Nik Prinčič

V okviru diplomskega dela je bilo implementirano in ovrednoteno delovanje vizualnega sledilnika na vgrajeni napravi Luxonis OAK-1. Izbran je bil sledilnik STARK, spada v družino sledilnikov, ki jih sestavljajo globoke nevronske mreže. Bolj specifično sledilnik uporablja arhitekturo transformer, ki je trenutno uporabljena v vseh najboljših vizualnih sledilnikih. Sledilnik je bilo potrebno rahlo predelati ter ga pretvoriti v OpenVINO format, ki omogoča uporabno na vgrajeni napravi. Poleg tega je bilo potrebno zasnovati cevovod, po katerem se podatki na napravi pretakajo. Z vsem naštetim smo dosegli, da lahko vgrajena naprava izvaja vse potrebne funkcije popolnoma avtonomno, vse kar potrebuje od gostiteljskega sistema (npr. osebni računalnik) je začetni omejitveni okvir tarče *(ang. bounding box)*, gostiteljskemu sistemu pa vrača vse naslednje omejitvene okvirje tarče. S tem smo dosegli to, da so performance sledenja neodvisne od gostiteljskega sistema.

**Ključne besede:** računalniški vid na vgrajenih napravah, DepthAI, vizualni sledilnik.

# Abstract

**Title:** Visual tracking on embedded devices

**Author:** Nik Prinčič

This sample document presents an approach to typesetting your BSc thesis using LaTeX. A proper abstract should contain around 100 words which makes this one way too short.

**Keywords:** embedded computer vision, DepthAI, visual tracker.

# Poglavje 1

# Uvod

Računalniški vid je področje, ki se v zadnjih letih zelo hitro razvija. Z napredkov v razvoju avtonomnih sistemov, kot so avtonomni avtomobili, droni, roboti in še mnogi drugi, in vse večjem številu IoT naprav opremljenih s kamero, se vedno bolj pojavlja želja po uporabi modernih pristopov na majhnih, manj zmogljivih napravah. Računalniškega vida zavzema kar nekaj področji, v tem delu smo se osredotočili na vizualno sledenje, natančneje sledenju posameznega objekta *(ang. single object tracking)*.

Vizualno sledenje je področje, ki se ukvarja z iskanjem in sledenjem objektov v videu. V tem delu smo se osredotočili na SOT, kjer je cilj slediti samo enemu objektu. Sledilniku je naprej potrebno podati BBOX tarče, kateri želimo slediti, nato pa sledilnik na podlagi prostorske, pri najbolj modernih pa celo časovno-prostorske informacije sledi želeni tarči in nam za vsako sličico vrne pripadajoč BBOX. V preteklosti so bil najbolj popularni tako imenovani klasični algoritmi (npr. KCF, MOSSE, ...), sedaj pa prevladujejo sledilniki temeljijo na nevronskih mrežah

Da bi lahko zagotovili, dobre performance, pri manjši porabi energije, so se začele razvijati namenske procesorske enote VPU, ki so optimizirane za izvajanje nevronskih mrež. Še posebej popularna je arhitektura transformer, ki je bila primarno razvita in uporabljena za razumevanje in generacijo teksta, v zadnjih letih pa je bila adaptirana na vizualne sledilnike.

V okviru te diplomske naloge smo se osredotočili na sledilnik STARK, ter vgrajeno napravo Luxonis OAK-1. Cilj naloge je bil sledilnik prilagoditi uporabi na tej napravi, ga prevesti v potreben format, ter ga umestiti v cevovod. V cevovodu je bilo potrebno tudi smiselno zasnovati, da v model pridejo pravilno oblikovani podatki.

Diplomsko delo je razdeljeno v N delov. V poglavju 2 bomo prestavili pregled področja. V poglavju 3 bomo opisali metodologijo, to zavzema hiter opis nevronskih mrež na splošno, arhitekture CNN in transformer, ter opis njune uporabe v vizualnih sledilnikih ter predstavitev izbranega sledilnika in vgrajene naprave. V poglavju 4 bomo opisali podrobnosti implementacije, kar vključuje opis prilagoditve model ter njegovo pretvorbo v pravilen format, opis implementacije cevovoda in opis 4 različnih načinov delovanja, ki smo jih pripravili. V poglavju 5 bomo prestavili rezultate evalvacije, osredotočili se bomo na primerjavo med sledilnikom, ki je bil pognan samostojno na vgrajeni napravi proti tistemu, ki je pognan na osebnem računalniku. Predstavili bomo tudi rezultate primerjave med dvema načinoma delovanja, robni *ang. edge mode* proti gostiteljskemu *ang. host mode* načinu delovanja. V poglavju 6 je zaključek, ki povzema ugotovitve, ter predstavi možne izboljšave za nadaljnji razvoj.

# Poglavje 2

# Pregled področja

V tem poglavju bomo pregledali dosedanje delo na področju vizualnega sledenja na splošno in na vgrajenih napravah. Na področju vizualnega sledenja je VOT test, eden najbolj uveljavljenih testov. V zadnjih letih prevladujejo sledilniki, ki uporabljajo arhitekturo transformer, to lahko tudi razberemo, če analiziramo rezultate zadnjega testa VOT2022. Iz članka, v katerem so bili objavljeni rezultati, lahko razberemo da 9 od najboljših 10 sledilnikov uporablja arhitekturo transformer, opazimo lahko tudi, da je kar 47% vseh testiranih sledilnikov uporabilo to arhitekturo.

Na področju vizualnega sledenja na vgrajenih napravah, je bilo objavljenih že nekaj del, a nobeno od njih ni uporabilo sledilnika na osnovi arhitekture transformer. V članku *Evaluation of Visual Tracking Algorithms for Embedded Devices*[4] so primerjali performance med 5 klasičnimi algoritmi, teste so izvedli na napravi Raspberry Pi 3 B v1.2 in ugotovili, da z uporabo sledilnika KCF dobijo najboljše razmerje med hitrostjo in natančnost. V članku *Real-Time Multiple Object Visual Tracking for Embedded GPU Systems*[2], so predstavili MOT *(ang. multiple object tracking)* na napravi Nvidia Jetson TX2, kjer so uporabili več stopenjsko arhitekturo, detektor (YOLOv3) in sledilnik (KCF) in dobili dobre rezultate.

# Poglavje 3

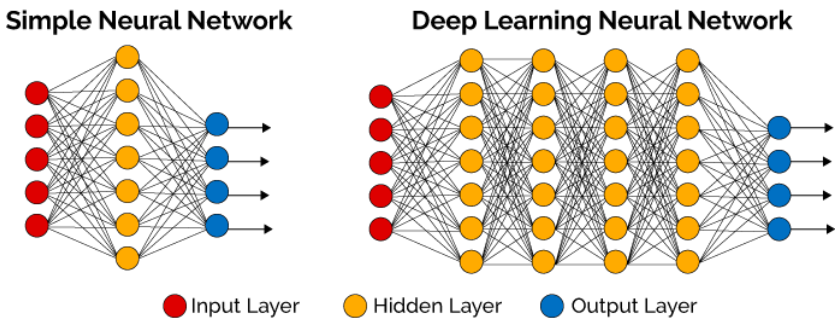
# Metodologija

Ker je bil cilj dela, preizkusiti možnost uporabe enega izmed najsodobnejših vizualnih sledilnikov na vgrajeni napravi, ki temelji na nevronskih mrežah, bom v tem poglavju predstavili delovanje nevronskih mrež, arhitekture CNN in transformer, ter opisali izbrani sledilnik in vgrajeno napravo.

## 3.1 Umetne nevronske mreže

Umetne nevronske mreže *(ang. Artificial Neural Network, ANN)* so vrsta modelov strojnega učenja, ki posnemajo delovanje bioloških nevronskih mrež *(ang. Biological Neural Network, BNN)*. Sestavljene so iz množice umetnih nevronov, ki so med seboj povezani z uteženimi povezavami in združeni v sloje. Sloje delimo na tri vrste - vhodni sloj *(ang. input layer)*, skrite sloje *(ang. hidden layers)* in izhodni sloj *(ang. output layer)*. Mreže z več kot enim skritim slojem imenujemo globoke nevronske mreže *(ang. Deep Neural Network, DNN)*.

Najpogosteje se za učenje nevronskih mrež uporablja algoritem vzvratnega razširjanja *(ang. backpropagation)*. Algoritem se izvaja v več iteracijah, pri katerih se s pomočjo kriterijske funkcije izračuna napaka, ki je razlika med želenim izhodom in dejanskim izhodom. Napako uporabimo za izračun gradienta, ki nam pove, kako se spremeni vrednost uteži, da se napaka zmanjša.



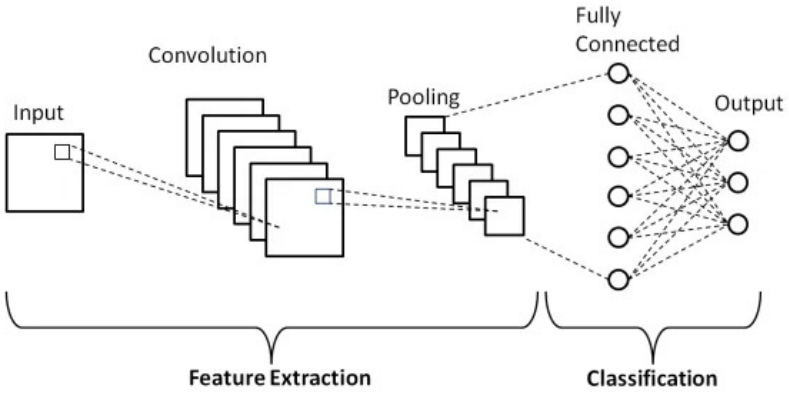

Slika 3.1: Primerjava zgradbe NN in DNN [5].

Kriterijsko funkcijo morem pravilno izbrati glede na vrsto problema, ki ga želimo rešiti. Za reševanje regresijskih problemov se najpogosteje uporablja srednjo kvadratno napako *(ang. Mean Squared Error, MSE)*, za klasifikacijske probleme pa najpogosteje kategorično križno entropijo *(ang. Categorical Cross-Entropy)* ali pa binarno križno entropijo *(ang. Binary Cross-Entropy)*. Za iskanje optimalnih vrednosti uteži se uporabljajo različni optimizacijski algoritmi, ki na podlagi gradientnega spusta *(ang. gradient descent)* iščejo minimum kriterijske funkcije. Med njimi sta najbolj znan stohastični gradientni spust *(ang. Stochastic Gradient Descent, SGD)* in Adam *(ang. Adaptive Moment Estimation)*.

## 3.2 Konvolucijske nevronske mreže

Konvolucijske nevronske mreže *(ang. Convolutional Neural Network)*, so vrsta umetnih nevronski mrež, ki se pogosto uporablja v nalogah računalniškega vida, kot so prepoznavanje objektov, klasifikacija slik, detekcija obrazov in drugo. CNN modeli so tipično sestavljeni iz več ponovitev konvolucijskih slojev *ang. pooling layers* in združevalnih slojev *ang. pooling layers*. Za zadnjim združevalnim slojem najpogosteje sledi nekaj polno-povezanih slojev. Poenostavljena arhitektura konvolucijske nevronske mreže je prikazana na sliki 3.2.

Namen konvolucijskih slojev je pridobivanje značilk *(ang. feature extrac-*




Slika 3.2: Poenostavljena arhitektura konvolucijkse nevronske mreže [1].

*tion)*, namen združevalnih slojev je zmanjševanje dimenzionalnosti podatkov, polno-povezani sloji pa so namenjeni preslikovanju pridobljenih značilk v končni izhod.

Konvolucijski sloji delujejo na principu matematične operacije konvolucije, ki je definirana na dveh funkcijah. Rezultat konvolucije nam pove, kako oblika ene spremeni obliko druge. Definirana je z enačbo:

$$(f * g)(t) = \int_{\tau=-\inf}^{\inf} f(\tau)g(t-\tau)d\tau \tag{3.1}$$

Kjer je $f$ predstavlja vhodno funkcijo, $g$ pa jedro konvolucije. Ker se pri obdelavi digitalnih podatkov pogosto uporablja diskretna konvolucija, je enačba 3.1 spremenjena v:

$$(f * g)[t] = \sum_{k=-\inf}^{\inf} f[k]g[n-k] \tag{3.2}$$

## 3.3 Arhitektura transformer

Transformer je arhitektura nevronske mreže, ki temelji na mehanizmu pozornosti, ki je bil predstavljen leta 2017 v članku *Attention is all you need* [6]. Primarno je bila arhitektur razvita za naloge procesiranja naravnega jezika *Natural language processing, NLP*, sejda pa postaja popularna tudi v drugih

domenah strojnega učenja, med drugim tudi v računalniškem vidu. Osnovna arhitektura, kji je bila predstavljena v [6], vključuje kodirni *(ang. encoder)* in dekodirni modul *(ang. decoder)*. Kodirnik je sestavljen iz dveh podslojev, dekodirnik pa iz treh.

### 3.3.1 Mehanizem pozornosti

Mehanizem pozornosti deluje na podlagi ključev *(ang. key, K)*, poizvedb *(ang. query, Q)* in vrednosti *(ang. value, V)*. Mehanizem iz matrike ključev in matrike poizvedb izračuna matriko pozornosti. Z matričnim množenjem matrike pozornosti in matrike vrednosti dobimo linearno kombinacijo vrednosti, ki predstavljajo izhod. Razlikujemo med samopozornostjo in medpozornostjo. O samopozornosti govorim, ko če vse tri vhodne parametre dobimo iz iste množice podatkov, pri medpozornosti pa poizvedbe pridobimo iz ene množice podatkov, ključe pa iz druge.

Vhodne vektorje $x_i, x_{i+1}, ..., x_n$ združimo v matriko $X_{n \times d_m}$, kjer $d_m$ predstavlja dimenzionalnost modela. Naučene parametre pa predstavljajo matrike $W^Q_{d_m \times d_m}$, $W^K_{d_m \times d_m}$ in $W^V_{n \times d_m}$. Iz navedenih matrik lahko izračunamo

$$\begin{aligned} Q &= XW^Q, \\ K &= XW^K, \\ V &= XW^V, \end{aligned} \tag{3.3}$$

Matrika pozornosti je definirana z enačbo:

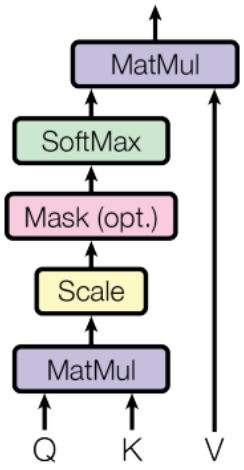
$$A(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_m}})V \tag{3.4}$$

kjer funkcija *softmax* spremeni vektor $n$ realnih števil v vektor verjetnostne porazdelitve.

Iz 3.3 in 3.4 lahko izračunamo končno izhodno vrednost mehanizma

$$S = AV \tag{3.5}$$

Ko se mehanizem pozornosti uporabi v dekodirnem modulu je potrebno določen del podatkov skriti. Bolj natačno, modelu je treba preprečiti, da bi lahko prejel podatke, kateri še niso bili napovedani. Skrivanje podatkov nam omogoča maska, ki določene vrednosti v matriki pozornosti $A$ nastavi na $-\infty$.



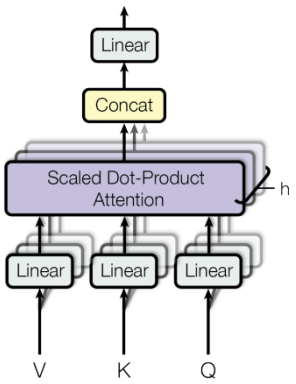

Slika 3.3: Diagram izračuna pozornosti [6].

### 3.3.2 Večglava pozornost

Da bi model lahko zajel več lastnosti vhodnih podatkov je potrebno mehanizem pozornosti nadgraditi. Večglava pozornost vzporedno izračuna več ločenih pozornosti (glav) in jih združi v končni rezultat. Vsaka glava se osredotoči na eno lastnost podatkov. Vsaka glava $i$ ima svoje matrike naučenih uteži, končni rezultat pa se pridobi s strnitvijo posameznih matrik pozornosti.

### 3.3.3 Vhodna vdelava in pozicijsko kodiranje

Preden vhodni podatki prispejo do kodirnega bloka jih je potrebno najprej pravilno predelati. Za ta namen se uporablja vdelava vhodnega niza v $d_m$ dimenzionalni latentni prostor *(ang. embedding space)*. Vhodna vdelava




Slika 3.4: Diagram izračuna večglave pozornosti [6].

*(ang. input embedding)* vhodne besede pretvori v vektorje, s tem model pridobi informacijo o pomenu posamezne besede. Pozicijsko kodiranje *(ang. positional encoding)* pa modelu pridobi informacijo o vrstnem redu besed v nizu. Da bi se izognili velikim vrednostim v pozicijskem kodiranju, se za kodiranje uporabljata sledeči funkciji

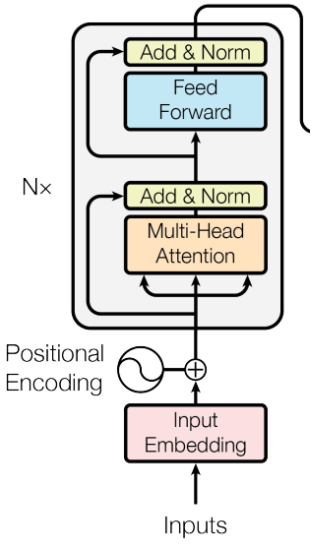
$$\begin{aligned} PE(pos, 2i) &= sin(pos/10000^{2i/d_m}) \\ PE(pos, 2i+1) &= cos(pos/10000^{2i/d_m}) \end{aligned} \tag{3.6}$$

v enačbi 3.5 predstavlja *pos* pozicijo besede v nizu, $i$ pa dimenzijo kodiranja, kar pomeni da ima vsaka dimenzija pozicijskega kodiranja pripadajočo sinusoidno vrednost.

Informacijo o pomenu in poziciji posamezne besede združimo tako, da matriki seštejemo.

### 3.3.4 Kodirni modul

Kodirni modul je sestavljen iz $N$ identičnih slojev, ki so sestavljeni iz dveh pod-slojev. Prvi pod sloj je več glava pozornost *(ang. multi-headed attention)* in polno povezane nevronske mreže *(ang. Feedforward neural network, FFN).*



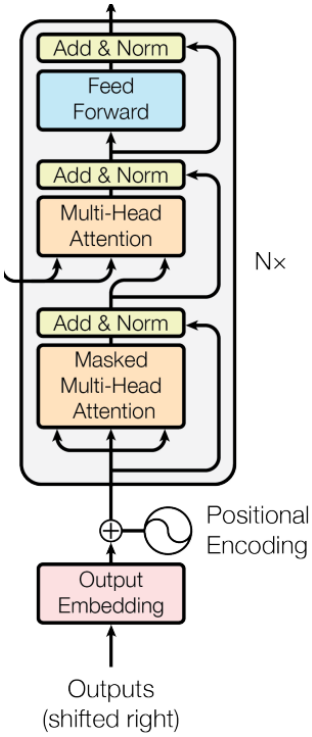

Slika 3.5: Diagram kodrinega modula [6].

### 3.3.5 Dekodirni modul

Dekodirni je sestavljen iz $N$ identičnih slojev, ki so sestavljeni iz treh podslojev. Prvi pod sloj je več glava pozornost z možnostjo maskiranja vhodnih podatkov. Drugi pod sloj je več glava pozornost, tretji sloj pa predstavlja polno povezana nevronska mreža.

## 3.4 Model STARK

Model STARK [7] spada v družino SOT *(ang. single object tracking)* sledilnikov. Primarno je sestavljen iz dveh arhitektur, konvolucijskih nevronskih mrež in arhitekture transformer, ki je bila prilagojena za vizualne sledilnike. Navdih za njegov nastanek je bil predhodni model za detekcijo DETER [**deter**]. Ena od novosti, ki so jo uvedli v tem modelu je uporaba časovne in prostorske komponente. Prostorska komponenta vsebuje informacijo o izgledu objekta, kateremu sledi. Časovna komponenta pa nosi informacijo o spremembi pozicije objekta skozi čas. Arhitektura, ki so jo predlagali




Slika 3.6: Diagram dekodirnega modula [6].

vsebuje tri ključne elemente: kodirni modul, dekodirni modul in napovedovalno glavo *(ang. predictio head)*. Kodirni modul kot vhod prejme trenutno sliko, ter matrica *(ang. template)*, ki se skozi čas dinamično posodablja. Moduli za samopozornost v kodirniku se povezave med vhodi in preko korelacije značilk. Ker se dinamična matrica, skozi čas, posodablja, lahko kodirnik zajame tudi prostorsko in časovno informacijo o objektu, ki mu sledimo.

Prednost tega sledilnika je v tem, da ne potrebuje kompleksnega pre in postprocesiranje. Ob inicializaciji prejem kot vhod sliko in omejitveni okvir tarče. Na podlagi teh dveh vhodnih podatkov se najprej izreže ROI, ki se uporabi za izračun matrice. Ob vsaki naslednji sličici, pa so vhodni podatki slika, začetna matrica in dinamična matrica, sledilnik pa vrne izračunani omejitveni okvir.

### 3.4.1 Arhitektura modela stark

Arhitekturo modela lahko razdelimo na tri glavne dele: *(1)* konvolucijska hrbtenica *(ang. convolutional backbone)*, *(2)* kodirni-dekodirni transformer in predikcijsko glavo za omejitveni okvir *(ang. bounding box prediction head)*. V nadaljevanju bomo predstavili vsak od teh delov.

#### Konvolucijska hrbtenica

Konvolucijsko hrbtenico sestavlja model ResNet [3], kateremu so odstranili zadnjo sekcijo in polno povezane sloje. Kot vhod prejme hrbtenica začetno matrico $z \in \mathbb{R}^{3\times H_z \times W_z}$ in iskalno regijo (iz slike izrezan ROI) $x \in \mathbb{R}^{3\times H_x \times W_x}$. Po prehodu čez hrbtenico dobimo dve matrici značilk $f_z \in \mathbb{R}^{C\times \frac{H_z}{s}\times \frac{W_z}{s}}$ in $f_x \in \mathbb{R}^{C\times \frac{H_x}{s}\times \frac{W_x}{s}}$.

# Članki v revijah

[2] Mauro Fernández-Sanjurjo, Manuel Mucientes in Víctor Manuel Brea. "Real-Time Multiple Object Visual Tracking for Embedded GPU Systems". V: *IEEE Internet of Things Journal* 8.11 (2021), str. 9177–9188. DOI: 10.1109/JIOT.2021.3056239.

https://www.overleaf.com/project/609ce2055f917cb2f776732e

# Članki v zbornikih

[3] Kaiming He in sod. "Deep residual learning for image recognition". V: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, str. 770–778.

[4] Ville Lehtola in sod. "Evaluation of Visual Tracking Algorithms for Embedded Devices". V: *Image Analysis*. Ur. Puneet Sharma in Filippo Maria Bianchi. Cham: Springer International Publishing, 2017, str. 88–97. ISBN: 978-3-319-59126-1.

[6] Ashish Vaswani in sod. "Attention is All You Need". V: 2017. URL: `https://arxiv.org/pdf/1706.03762.pdf`.

[7] Bin Yan in sod. "Learning spatio-temporal transformer for visual tracking". V: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, str. 10448–10457.

# Celotna literatura


[1] Sai Balaji. *Binary Image classifier CNN using TensorFlow*. URL: https://medium.com/techiepedia/binary-image-classifier-cnn-using-tensorflow-a3f5d6746697 (pridobljeno 2023).

[2] Mauro Fernández-Sanjurjo, Manuel Mucientes in Víctor Manuel Brea. "Real-Time Multiple Object Visual Tracking for Embedded GPU Systems". V: *IEEE Internet of Things Journal* 8.11 (2021), str. 9177–9188. DOI: 10.1109/JIOT.2021.3056239.

[3] Kaiming He in sod. "Deep residual learning for image recognition". V: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, str. 770–778.

[4] Ville Lehtola in sod. "Evaluation of Visual Tracking Algorithms for Embedded Devices". V: *Image Analysis*. Ur. Puneet Sharma in Filippo Maria Bianchi. Cham: Springer International Publishing, 2017, str. 88–97. ISBN: 978-3-319-59126-1.

[5] Nisarg Patel. *What Is Deep Learning?* URL: https://medium.com/@nnpatel4583/what-is-deep-learning-4daa22ceea4e (pridobljeno 2023).

[6] Ashish Vaswani in sod. "Attention is All You Need". V: 2017. URL: https://arxiv.org/pdf/1706.03762.pdf.

[7] Bin Yan in sod. "Learning spatio-temporal transformer for visual tracking". V: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, str. 10448–10457.