Vpliv parametrov barvnega modela pri robotskem ravnanju tekstila

Peter Nimac[1,2], **Andrej Gams**[1]

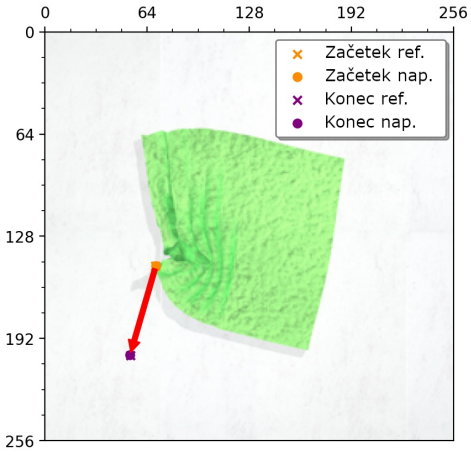
[1]*Laboratorij za humanoidno in kognitivno robotiko, Odsek za avtomatiko biokibernetiko in robotiko, Inštitut "Jožef Stefan", Jamova cesta 39, 1000 Ljubljana*
[2]*Mednarodna podiplomska šola Jožefa Stefana, Jamova cesta 39, 1000 Ljubljana*
E-pošta: peter.nimac@ijs.si

Color model parameter impact on robot textile flattening

Using robots to handle textile elements is an increasingly active, although still largely unexplored research area. This is due to complexity of actions, which are needed to successfully handle textile elements. Properties of textile, most notably high deformability, self-collision and self-occlusion cause difficulty in accurately determining the state of the textiles. Due to the considerable variability in the shape and size of planar, non-rigid objects, we have addressed this challenge by using deep learning methods. In this work, we demonstrate a vision-to-motion DNN (Deep Neural Network) trained to straighten a single crumpled corner on a square piece of fabric that was deformed and then flattened inside a simulated environment. Our model was trained in a simulated environment to correctly recognise a pair of points which define the beginning and the end to linear flattening motion. For this simplified example, our trained model was able to achieve good results with an average error of 3.5 mm in determining the grab point position and an average error of 0.6 mm in determining the drop point position.

1 Uvod

Manipulacija tekstilov s pomočjo robotov ostaja zahteven izziv. Ta zahtevnost izhaja iz težko predvidljivega obnašanja tekstila zaradi stalnega spreminjanja lastne oblike. Zaradi tega težavnega predvidevanja se trenutno aktivno raziskuje rabo različnih pristopov strojnega učenja manipulacije tkanin [1]–[8]. Algoritmi spodbujevalnega učenja [1]–[3] omogočajo rabo enega samega algoritma, ki se lahko nauči vrsto različnih nalog, vendar mnogokrat na račun nerazložljivosti in možne občutljivosti na nepoznane primere [9]. Nekatere od teh težav se da nasloviti z rabo algoritmov učenja s posnemanjem [4]. Pri učenju s posnemanjem potrebujemo veliko število demonstracij, kar ni enostavno izvedljivo v primerih, kjer za učenje potrebujemo velike učne množice. Za dolgoročno načrtovanje poteka gibov je smiselno stanja tkanin (bodisi kot slike bodisi na drug način) preslikati v skrit prostor povezanih stanj (angl. latent space) [5]. Prek teh stanj lahko nato načrtujemo stanja in gibe za prehode med njimi, da tkanino postavimo v končno stanje. Izpeljava celotnih potekov od zmečkanega do zloženega tekstila je bila uspešno prikazana na dvo-ročnih sistemih [6]–[8].



Slika 1: Napovedani točki za začetek in konec giba ravnanja glede na vhodno sliko s spremenjenim barvnim tonom. Križci označujejo referenčno začetno točko giba (oranžna) in referenčno končno točko (vijolična). Piki, ki se na danem primeru prilegata križcema, označujeta točki, ki jih je napovedal model.

Ena izmed pod-nalog v teh procesih je izravnava zmečkanih tekstilov, ki se lahko izvede z različnimi pristopi.

V robotskih aplikacijah modele nevronskih omrežij (angl. *deep neural network* - DNN) pogosto učimo v simuliranih okoljih, bodisi deloma bodisi v celoti. Poglavitna prednost tega pristopa je, da omogoča pohitritev učenja in prepreči obrabo in možnosti poškodbe opreme, predvsem pa simulirano okolje lažje nadzorujemo, da ostanejo parametri učenja ponovljivi [1]. Toda, težava simulacije je, da je ta le približek realnosti, saj zaenkrat še ne poznamo simulatorja, ki bi v popolnosti posnemal vse parametre in fizikalne pojave iz resničnosti. Zaradi tega velja, da modeli, ki so dobro naučeni v simulaciji, tudi dobro učinkujejo v simulaciji. Zaradi omenjenih razlik med simuliranim in stvarnim okoljem uspešnost teh modelov upade, ko jih uporabimo v okolju, ki smo ga simulirali med učenjem [10]. Za uspešen prenos modela v realni svet, se poslužimo pogosto uporabljene tehnike, vpeljave naključne porazdelitve v učno množico ali domeno (angl. *domain randomisation*). Vpeljavo naključne porazdelitve lahko v grobem delimo na vizualno naključno porazdelitev in naključno porazdelitev dinamike [11]. Z vnašanjem faktorja naključnosti v tiste podatke oz. parametre, ki niso pomembni za delovanje modela (barva, manjša

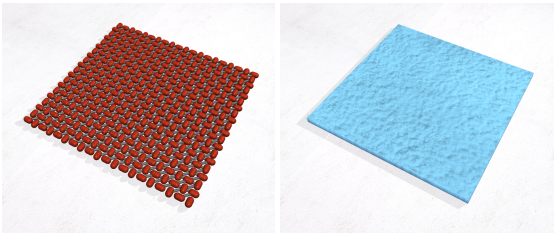
variabilnost v elastičnosti in debelini, itd.) zmanjšamo razlike med simulacijo in resničnostjo, ko se približamo porazdelitvi realnega sveta [12].

Naš model smo učili, da izravna kvadraten kos tkanine, pri čemer smo za boljšo posplošitev modela barvo tkanine v učni množici porazdelili naključno. Na tej točki smo izbrali samo barvno naključno porazdelitev, saj lahko z izbranim pristopom naključno porazdelitev barve vnesemo na že obstoječih zbirkah podatkov.

V tem prispevku demonstriramo uporabo metode za vnos naključne porazdelitve v barvi s spreminjanjem enega samega parametra. Uporabljen model, ki smo ga predhodno opisali v [13], je zmožen napovedati začetno in končno točko linearnega giba. S tem gibom robot nato izravna simulirano krpo, kot to prikazuje slika 1. Preverili smo tudi razliko v uspešnosti modela, ko je ta učen samo na slikah, kjer je barva krpice konstantna in ko je ta učen na slikah, kjer je barva naključna.

2 Zbirka podatkov

Za uspešno učenje modela potrebujemo primerno zbirko podatkov sestavljeno iz slik in ujemajočih gibov. Podatke smo zbrali znotraj simulatorja MuJoCo [14]. Tkanina je bila v simulatorju predstavljena kot krpica kvadratne oblike s stranico $a_s = 28$ cm in debelino $a_t = 5$ mm. V simulatorju je bila opisna kot mreža fleksibilno povezanih kapsul z rastrom 20×20, kot je to prikazano na sliki 2a.

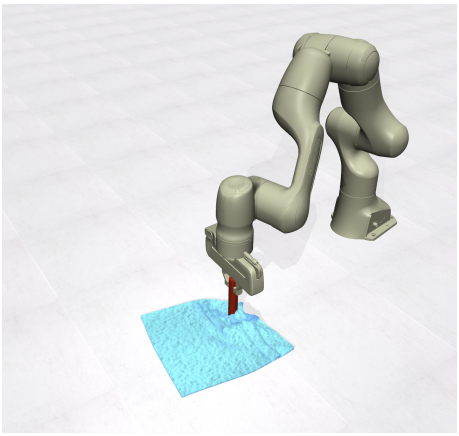


(a) (b)

Slika 2: Raster vozlišč v obliki kapsul, ki v simulatorju predstavljajo model krpice. Kapsule na sliki 2a so dejanski trdni objekti katere v simuliranjem okolju premikamo z robotom. Za končno vizualizacijo na sliki 2b sta oblika in tekstura krpe napeti okoli kapsul kot opno, ki na samo manipulacijo nimata vpliva.

Zbirko smo sestavili iz skupaj 4000 simulacij deformiranja. Kot v našem prejšnjem delu [13] je bila krpa ob začetku vsakega simulacijskega cikla postavljena v splošno izravnan položaj, na kar smo z robotom kot krpice potisnili proti sredini za naključno razdaljo med 7 cm in 25 cm ter pod naključnim kotom med 20° in 70° (slika 3). Po končanem deformiranju smo zajeli sliko stanja ter par točk, kjer je robot začel in zaključil deformacijski gib. Tekom zbiranja podatkov je barva krpe ostala konstantna.

Naključno porazdelitev barve krpice smo v zbirko vnesli z naključnim spreminjanjem vrednosti parametra barvnega tona oziroma barvitosti (angl. hue) med vrednostma 0°in 360°. Barvni ton je parameter v barvnih modelih HSV (angl. za hue, saturation, value) in HSL (angl. za



Slika 3: Simulacija deformacije z linearnim gibom.

hue, saturation, lightness), ki predstavljata alternativo barvnemu modelu RGB [15], [16]. Ta modela omogočata, da barvo slikovnih točk spreminjamo samo s parametrom barvnega tona. Če bi slikovnim točkam želeli spremeniti barvo po barvnem modelu RGB, bi za to morali spreminjati tri parametre, to so intenzivnosti kanalov rdeče R, zelene G in modre barve B. Sprememba barvnega tona vpliva samo na slikovne pike, kjer vrednosti komponent R, G in B niso enake. To pomeni, da bela podlaga s temnimi pikami ostane nespremenjena pri katerikoli spremembi vrednosti barvnega tona. Z dodatnim spreminjanjem parametrov nasičenosti in intenzivnosti, bi lahko do neke mere naključno porazdelili tudi svetlobne pogoje slike.

Pretvorba med barvnima prostoroma RGB in HSV poteka po poznanem postopku [15]–[17]:

Za komponente R, G in B velja

$$R, G, B \in [0, 1], \tag{1}$$

kjer za posamezno slikovno točko iščemo maksimalno vrednost V in minimalno vrednost X

$$V = \max(R, G, B), \tag{2}$$

$$X = \min(R, G, B). \tag{3}$$

Nasičenost S izračunamo kot

$$S = \begin{cases} 0, & V = 0; \\ \dfrac{V - X}{V}, & \text{sicer} \end{cases} \tag{4}$$

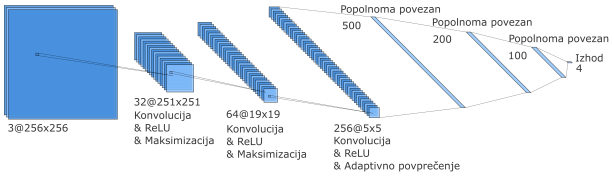
in barvni ton H izračunamo kot

$$H = \begin{cases} 0, & V = X; \\ 60°(\dfrac{G - B}{V - x} \mod 6), & V = R; \\ 60°(\dfrac{B - R}{V - x} + 2), & V = G; \\ 60°(\dfrac{R - G}{V - x} + 4), & V = B. \end{cases} \tag{5}$$

Za sivinske vrednosti $S = 0$ barvni ton H ni definiran.

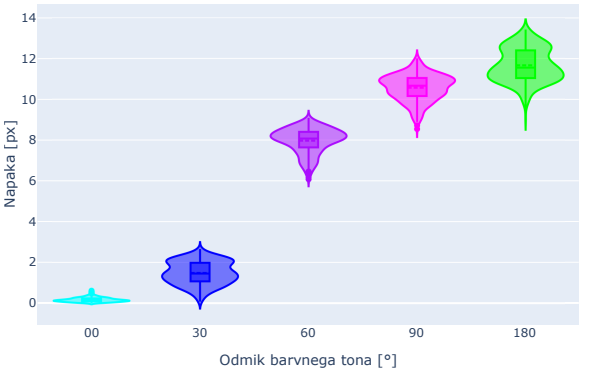
3 Globoko nevronsko omrežje slika v gib

Učenje smo nadaljevali na istem globokem konvolucijskem nevronskem omrežju (slika 4) kot v našem prejšnjem delu [13]. Omrežje temelji na osnovi [18], kjer so Pahič idr. slikovno informacijo prevedli v robotski gib za prepisovanje z roko napisanih števil.Kot vhod v nevronsko omrežje uporabimo barvno sliko RGB v obliki $3 \times 256 \times 256$ in na njej izvedemo operacije konvolucije, popravljanja linearnih enot (angl. rectified linear unit - ReLU) in združevanja slojev z maksimiranjem (angl. max-pooling). Ta postopek ponovimo dvakrat, v tretjem koraku pa maksimiranje nadomestimo z adaptivnim povprečenjem. Nato vrednosti preoblikujemo v vektorsko obliko treh popolnoma povezanih slojev s štiri dimenzijskim izhodom na koncu omrežja. Izhod predstavlja par ravninskih koordinat začetka in konca izravnalnega giba.

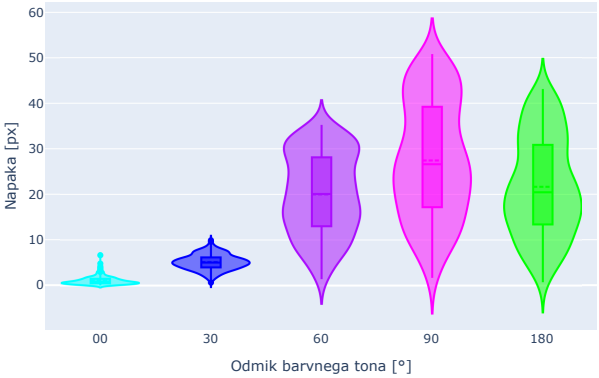


Slika 4: Zgradba globokega nevronskega omrežja slika v gib.

4 Rezultati in diskusija



(a) Napaka v napovedovanju končne točke giba.



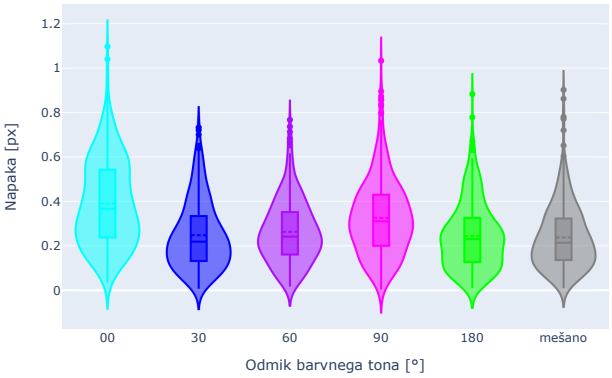
(b) Napaka v napovedovanju začetne točke giba.

Slika 5: Napaka pri posameznih spremembah barvnega tona, ko je omrežje naučeno na enobarvnih slikah.

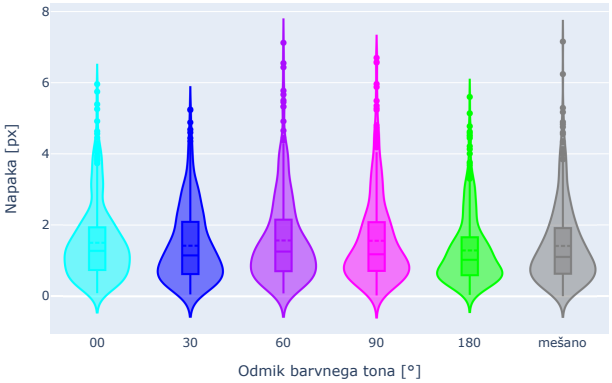
Sprva smo preverili vpeljavo naključnega nabora barv na modelu, ki je bil učen na izvornih 4000 vzorcih enobarvne krpice (slika 5). Za namene testiranja smo uporabili pet različic testne množice z 200 vzorci. Prva testna množica je bila sestavljena iz vzorcev slik zajetih iz simulatorja, vzorcem v drugi množici smo vnesli odmik od izvornega barvnega tona za 30°, tretji za 60°, četrti za 90° in peti za 180°. Tako smo preverili faktor preprileganja modela glede na barvo krpice, saj se z večanjem barvne razlike matematično veča tudi razdalja med izvorno barvo in novo barvo. Ta pojav lahko opazimo na sliki 5, kjer se napaka nevronskega omrežja veča v odvisnosti od kotnega odmika barvnega tona. Odmik barvnega tona za 30°je imel še sprejemljivo napako, povečano za $5,1$ px pri napovedovanju začetne točke premika in $1,5$ px pri napovedovanju končne točke premika. Pri večjih odmikih sta se povečala tako napaka kot tudi raztros, predvsem pri napovedovanju začetne točke premika. Raztros napake je pri napovedi končne točke giba ostal konsistenten. Predvidevamo da zato, ker se slikovna informacija okoli referenčne končne točke ni spreminjala.

4.1 Uspešnost modela po učenju na barvno-raznoliki učni množici

Model smo nato preverili z učenjem na spremenjeni učni množici, z naključno porazdelitvijo barve (slika 6). Uspešnost modela smo preverili na istih testnih množicah kot v prejšnjem primeru.



(a) Napaka v napovedovanju končne točke giba.



(b) Napaka v napovedovanju začetne točke giba.

Slika 6: Izboljšava uspešnosti modela po učenju na barvno raznolikih slikah.

Dodali smo še šesto testno množico, pri kateri smo na enak način vnesli naključno porazdelitev kot za novo učno množico. Po vpeljavi naključne porazdelitve barve v učno in testno množico, je povprečna napaka v napovedi modela $1,46$ px pri napovedovanju začetne točke in $0,26$ px pri napovedovanju končne točke. To v metričnih enotah predstavlja napako $3,5$ mm in $0,6$ mm. Opazimo tudi, da se napaka pri vseh barvah enakomerno razporedi.

5 Zaključek

Z izbranim pristopom smo demonstrirali, da lahko za obstoječe zbirke podatkov v podobnih primerih naknadno in efektivno vnesemo naključno porazdelitev domene. Naš primer je bil deloma poenostavljen, saj nam ni bilo potrebno ločevati med subjektom interesa (krpica) in ozadjem. Kljub poenostavljenemu primeru smo prikazali, da se s primerno vpeljavo naključne porazdelitve v učno množico močno poveča robustnost modela, kljub temu, da smo naključno porazdelitev vnesli v že obstoječo učno množico. Za kompleksnejše primere, kjer je ločitev potrebna, moramo uporabiti naprednejše pristope. Kjer je to možno, lahko uporabimo slikovno razčlenjevanje oz. segmentacijo, denimo v primerih, kjer je na eni sliki sočasno več različnih krp. S slikovnim razčlenjevanjem bomo tako vsaki krpi posebej spreminjali vizualne parametre. Razčlenjevanje tudi omogoča ločitev tekstilov od poljubnega ozadja, kateremu bo tudi možno vnesti naključno porazdelitev. V poštev pride tudi uporaba slikovnih generatorjev za dodajanje naključnih vzorcev. Z naključnim spreminjanjem parametrov nasičenosti in intenzivnosti pa bomo posnemali tudi različne svetlobne pogoje.

Literatura

[1] J. Matas, S. James in A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation", *Conference on Robot Learning*, PMLR, 2018, str. 734–743.

[2] Y. Tsurumine, Y. Cui, E. Uchibe in T. Matsubara, "Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation", *Robotics and Autonomous Systems*, let. 112, str. 72–83, 2019.

[3] Y. Wu, W. Yan, T. Kurutach, L. Pinto in P. Abbeel, "Learning to Manipulate Deformable Objects without Demonstrations", *arXiv:1910.13439 [cs]*, mar. 2020, arXiv: 1910.13439.

[4] D. Seita, A. Ganapathi, R. Hoque in sod., "Deep Imitation Learning of Sequential Fabric Smoothing Policies", *CoRR*, let. abs/1910.04854, 2019.

[5] M. Lippi, P. Poklukar, M. C. Welle in sod., "Latent Space Roadmap for Visual Action Planning of Deformable and Rigid Object Manipulation", *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, str. 5619–5626.

[6] A. Doumanoglou, J. Stria, G. Peleka in sod., "Folding Clothes Autonomously: A Complete Pipeline", *IEEE Transactions on Robotics*, let. 32, str. 1461–1478, 2016.

[7] Y. Avigal, L. Berscheid, T. Asfour, T. Kroger in K. Goldberg, "SpeedFolding: Learning Efficient Bimanual Folding of Garments", *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, str. 1–8, 2022.

[8] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal in D. Held, "FabricFlowNet: Bimanual Cloth Manipulation with a Flow-based Policy", *Proceedings of the 5th Conference on Robot Learning*, A. Faust, D. Hsu in G. Neumann, ur., zbirka Proceedings of Machine Learning Research, zv. 164, PMLR, 2022, str. 192–202.

[9] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup in D. Meger, "Deep Reinforcement Learning that Matters", *AAAI Conference on Artificial Intelligence*, 2017.

[10] F. Muratore, F. Ramos, G. Turk, W. Yu, M. Gienger in J. Peters, "Robot Learning From Randomized Simulations: A Review", *Frontiers in Robotics and AI*, let. 9, 2022.

[11] W. Zhao, J. P. Queralta in T. Westerlund, "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey", *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 737–744, 2020.

[12] J. Tobin, "Real-World Robotic Perception and Control Using Synthetic Data", University of California, Berkeley, 2019.

[13] P. Nimac, M. Mavsar in A. Gams, "Cloth Smoothing Simulation with Vision-to-Motion Skill Model", *Zbornik enaintridesete mednarodne Elektrotehniške in računalniške konference*, Društvo Slovenska sekcija IEEE, 2022.

[14] E. Todorov, T. Erez in Y. Tassa, "MuJoCo: A physics engine for model-based control", *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, str. 5026–5033.

[15] A. R. Smith, "Color Gamut Transform Pairs", *SIGGRAPH Comput. Graph.*, let. 12, št. 3, str. 12–19, 1978.

[16] G. H. Joblove in D. Greenberg, "Color Spaces for Computer Graphics", *SIGGRAPH Comput. Graph.*, let. 12, št. 3, str. 20–25, 1978.

[17] H. Levkowitz in G. T. Herman, "GLHS: A Generalized Lightness, Hue, and Saturation Color Model", *CVGIP: Graph. Models Image Process.*, let. 55, št. 4, str. 271–285, 1993.

[18] R. Pahič, B. Ridge, A. Gams, J. Morimoto in A. Ude, "Training of deep neural networks for the generation of dynamic movement primitives", *Neural networks : the official journal of the International Neural Network Society*, let. 127, str. 121–131, 2020.