DETEKCIJA PUŠČIC PRI KLASIČNEM PIKADU

MATIC ZGONC, BORUT BATAGELJ

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Ljubljana, Slovenija
mz4144@student.uni-lj.si, borut.batagelj@fri.uni-lj.si

Sinopsis V članku je opisan razvoj sistema za detekcijo puščic pri klasičnem pikadu, z uporabo metod računalniškega vida, ki so implementirane v odprtokodni knjižnici OpenCV. Cilj je bil razviti sistem z visoko natančnostjo, ki bo enostaven za uporabo in cenovno dostopen. Na začetku je opisan postopek kalibracije tarče, sledi iskanje točke udarca in izračun točk in na koncu še uporabljene prilagoditve pri razvoju sistema.

Ključne besede:
OpenCV,
računalniški vid,
pikado,
homografija,
kalibracija

Univerzitetna založba Univerze v Mariboru

DOI https://doi.org/10.18690/um.feri.4.2023.11
ISBN 978-961-286-720-1

Arrow Detection for Classic Darts

Matic Zgonc, Borut Batagelj
University of Ljubljana, Faculty of Computer Science and Informatics, Ljubljana, Slovenia
mz4144@student.uni-lj.si, borut.batagelj@fri.uni-lj.si

Abstract This paper describes the development of a system for arrow detection in classic darts using computer vision methods in the open-source library OpenCV. The goal was to develop a highly accurate system that is easy to use and affordable. At the beginning, the procedure of target calibration is described, followed by the search for the hit point and the calculation of the score, and finally a description of the adjustments used in the development of the system.

Keywords: OpenCV, computer vision, darts, homography, calibration

https://doi.org/10.18690/um.feri.4.2022.11 DOI
978-961-286-720-1 ISBN

University of Maribor Press

1 Uvod

Pikado je šport, ki je pri nas in po svetu vedno bolj priljubljen. V zadnjih letih se opaža velika rast profesionalnih igralcev, gledalcev in denarnih nagrad. Na profesionalnih tekmovanjih se največkrat uporablja klasična, pri amaterskih igralcih pa elektronska tarča. Najpogostejši razlog je, da elektronska tarča omogoča štetje rezultata in podpira različne vrste iger, medtem, ko moramo pri klasični vse štetje opraviti ročno. To nas je vodilo k razvoju sistema, za avtomatsko zaznavanje puščic in štetje točk.

Na trgu že obstajajo rešitve, ki rešujejo ta problem. Najbolj znana in razširjena v naši okolici je Scolia (Scolia, 2023). Deluje na principu uporabe več kamer iz različnih zornih kotov, ki služijo zaznavanju puščic. Njegova glavna slabost je ročna kalibracija sistema, ki jo je potrebno ponoviti ob vsaki spremembi stanja tarče oz. pozicije kamer. Pri izdelavi našega sistema, smo si zadali implementiranje avtomatske kalibracije, ki jo lahko ponovimo kadarkoli med njegovim delovanjem.

Izgradnjo sistema smo razdelili na 3 sklope, in sicer kalibracijo tarče, iskanje točke udarca puščice, ter pridobivanje rezultata posameznega meta.

2 Izgradnja sistema

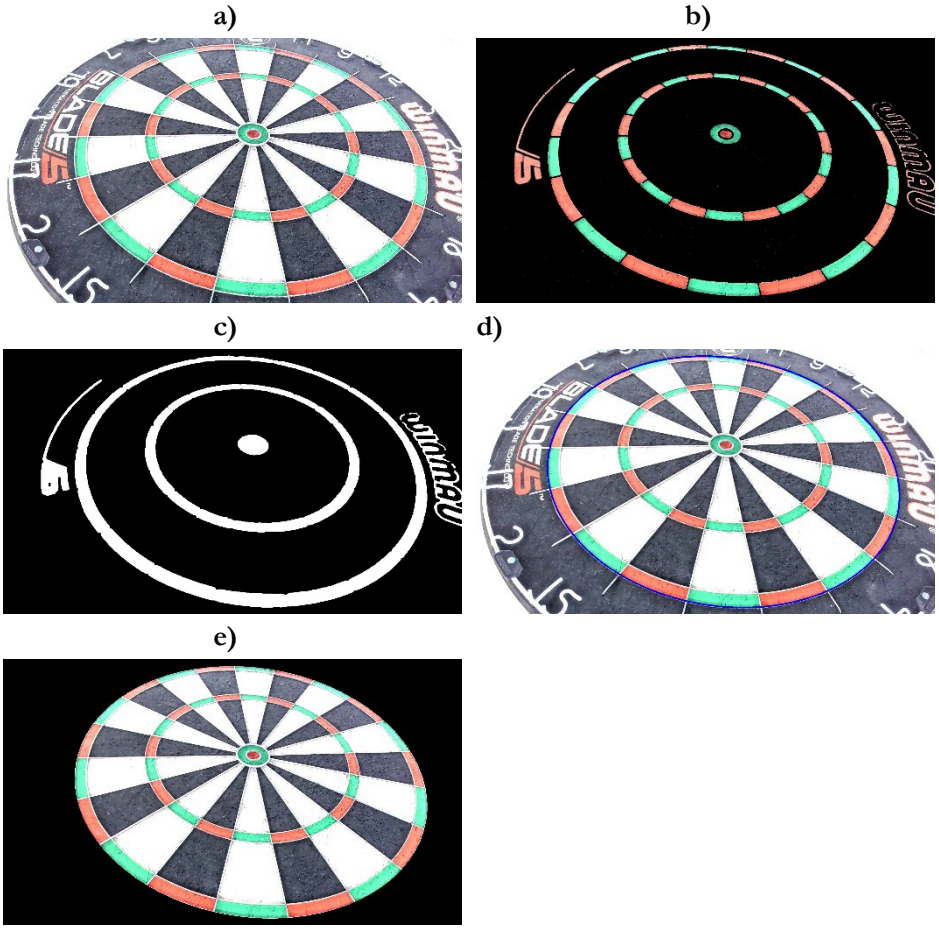
2.1 Kalibracija tarče

Pri kalibraciji tarče je cilj pridobitev transformacijske matrike. Ta nam vhodno sliko, na kateri je tarča v obliki elipse, pretvori v pravilno okroglo obliko. Uporabimo jo tudi pri pridobivanju točke udarca in rezultata meta.

Transformacijsko matriko pridobimo iz 4 točk, ki se na igralni površini tarče nahajajo 90° ena od druge. Pri pridobivanju transformacijskih točk, smo si pomagali s HSV (hue, saturation, value) barvnim prostorom, ter morfološkimi operacijami (Chhikara, 2022; Soille, 1999).

Prvi korak je iskanje elipse, ki obkroža igralno površino. Pridobimo jo z uporabo dveh HSV mask, eno za zeleno barvo in drugo za rdečo, ki jih združimo. Posamezna maska, na sliki ohrani slikovne točke, ki so v njenih mejah, ostale zavrže in jih označi z 0. Tako pridobimo dva kolobarja z zelenimi in rdečimi polji, ki pa med seboj niso

povezani. Z uporabo morfoloških transformacij dilacije in erozije pridobimo kolobar, ki je primeren za iskanje obrisov (angl. contours). Pridobljen obris že kaže podobo elipse, ki pa ni čisto pravilne oblike in vsebuje manjše nepravilnosti. Funkcija *fitEllipse* iz OpenCV paketa (Laganiere, 2017), po metodi najmanjših kvadratov izračuna elipso, ki se najbolj prilega na dan nabor 2D točk. Z izračunano elipso na koncu iz vhodne slike izločimo zunanje območje elipse. S tem odstranimo morebitne motnje, ki bi lahko oteževale nadaljnjo kalibracijo. Koraki do pridobitve elipse, so prikazani na Sliki 1.



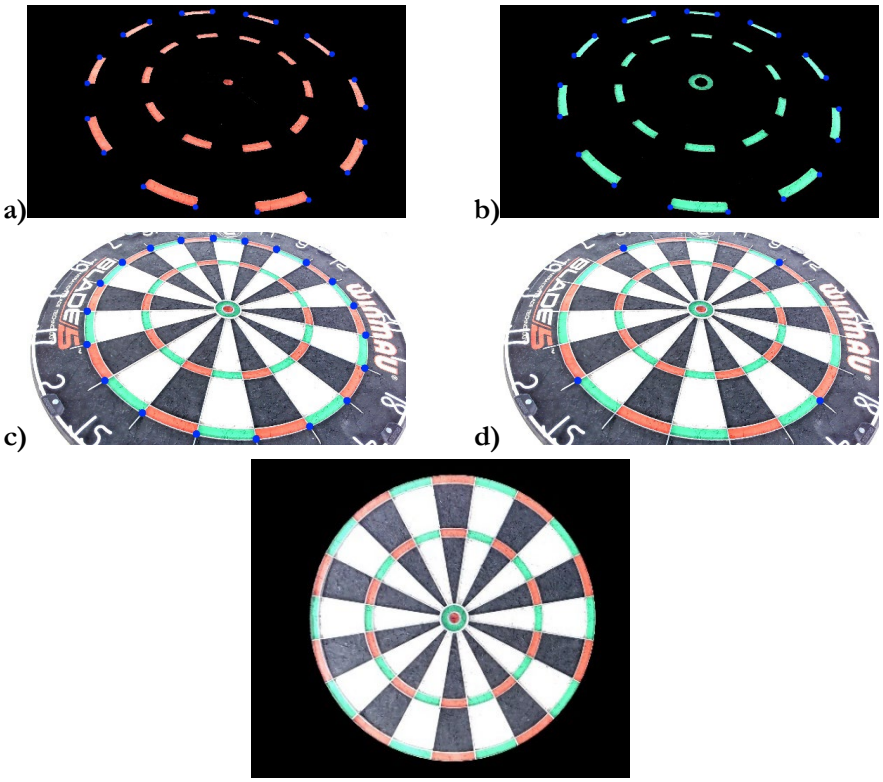
Slika 1: Vhodna slika (a), HSV maski (b), morfološke operacije (c), poiskana elipsa (d), izrezana slika tarče iz okolice (e)

Vir: lasten.

Naslednji korak je iskanje točk, ki razmejujejo sektorje na zunanjem krogu. Ponovno smo uporabili HSV maske, vendar tokrat ločeno. Tako pridobimo točke, ki razmejujejo zelene in rdeče sektorje. Za vhod v transformacijsko matriko izberemo štiri točke, ki so med seboj oddaljene 5 sektorjev.

Pred izvedbo transformacije, izračunamo še homografsko matriko (angl. Homography) z uporabo funkcije *findHomography*, ki mapira točke med dvema ravninama.

Zadnji korak je izvedba transformacije z uporabo funkcije *warpPerspective*. Ta na vhod dobi homografsko matriko, vhodne točke, ter ciljne točke, ki predstavljajo točke na poravnani tarči. Rezultat transformacije je kalibrirana tarča. Slika 2 prikazuje iskanje točk in kalibrirano sliko.



Slika 2: Posamezni HSV maski (a, b), točke zunanjih sektorjev (c), izbrane točke za transformacijo (d), kalibrirana slika (e)

Vir: lasten.

2.2 Iskanje točke udarca

Iskanje puščice smo izvedli s principom absolutne razlike med dvema slikama. Prva je posneta pred metom, druga pa po tem, ko puščica prileti v tarčo. Če so svetlobni pogoji v obeh primerih podobni, nam po izračunu razlike, ostane na sliki le puščica brez okolice. Da lahko poiščemo točko udarca, sliko najprej binariziramo, ter nato poiščemo obris puščice. Zaradi oblike puščice, točka udarca predstavlja najnižje ležečo točko na njej. Dobljene koordinate preslikamo v kalibrirano ravnino, da lahko izračunamo rezultat meta. Slika 3 prikazuje korake od izračuna absolutne razlike, do preslikane točke udarca.

Slika 3: Koraki do pridobitve točke udarca.
Vir: lasten.

2.3 Izračun točk

Tarča je razdeljena na 20 sektorjev, kar pomeni, da je vsak širok 18°. Z izračunanim kotom med preslikano točko udarca in središčem kalibrirane tarče, dobimo vrednost zadetih točk. Z uporabo kota lahko pridobimo le točke od 1 do 20, ne pa tudi vrednosti sredinskih krogov, ki sta vredna 25 in 50 točk, ter dvojnega in trojnega multiplikatorja. Ker so dimenzije tarč standardizirane, lahko za izračun teh vrednosti uporabimo razdaljo od točke udarca, do središča tarče. Za izračun končnih točk, enostavno pomnožimo vrednost zadetih točk z multiplikatorjem.

3 Prilagajanje sistema

V procesu razvoja smo izvedli različne prilagoditve, da bi izboljšali delovanje sistema in dosegali čim boljšo uspešnost zaznavanja. Primerjali smo različne osvetlitve tarče, različne kote namestitve kamere, uporabo kamer različnih kakovosti in namestitev kamere na različnih lokacijah. Pri vsaki prilagoditvi smo izdelali testne množice slik (Tabela 1), in na podlagi tega izbrali ustrezno rešitev, ki prinaša najboljše rezultate.

Tabela 1: Specifikacija testnih množic

Prilagoditev	Velikost testne množice
Osvetlitev tarče	111
Kot namestitve kamere	120
Različni kameri	117
Različne postavitve kamere	123

Pri osvetlitvi tarče smo preizkušali uspešnost zaznavanja v naravni osvetlitvi prostora in z uporabo led trakov. Za ta namen smo izdelali leseno ohišje za tarčo, okoli katerega smo namestili osvetlitev. Uporabili smo dve konfiguraciji led trakov. V prvi smo uporabili trak srednje moči, naravne barve svetlobe, v drugi pa smo poleg dodali še močnejši trak hladne bele svetlobe.

Ugotovili smo, da je za zaznavanje najboljša uporaba katerekoli konfiguracije led trakov, saj so bile vse puščice dobro vidne in pravilno zaznane, medtem, ko pri naravni osvetlitvi več kot 15% puščic sistem ni zaznal. Največ težav je bilo zaradi senc in pri puščicah v črnih poljih. Uporaba led trakov je te težave odpravila.

Pri naslednji prilagoditvi smo iskali kot, pod katerim kamera najboljše zaznava puščice. Identične kamere smo namestili na isto os, ki je bila pravokotna na tarčo. Preizkusili smo tri kote, in sicer 40°, 52°, in 60°. Izkazalo se je, da v uporabljeni testni

množici, sistem najboljše zaznava puščice pri kotu 40°, kjer je bilo 5 prekritih puščic. Na ostalih dveh pozicijah, je prišlo do 8 primerov, kjer je peresce prekrilo konico druge puščice in onemogočilo zaznavo. Z večanjem kota, se povečuje zakritost vidnega polja zaradi peresca.

Preverili smo tudi, če kamera različne kakovosti in cenovnega razreda, vpliva na uspešnost. Izbrali smo spletni kameri, z ločljivostjo vsaj 1080p, saj dosegajo najboljše razmerje med kvaliteto in ceno. Prva je znamke Logitech, s senzorjem velikosti 2MP, ki omogoča zajemanje slik do ločljivosti 1920x1080 slikovnih točk in stane okoli 50€. Druga je znamke EKACOM, s 5MP senzorjem in najvišjo ločljivostjo 2560x1440 slikovnih točk. Njena cena je bila v času nakupa dobrih 25€. Pri testiranju smo izklopili samodejno ostrenje, ostale nastavitve pa so ostale privzete.
Uspešnost na testni množici je bila pri kameri EKACOM 98 %, pri Logitech pa 96 %. Menimo, da bi z manjšimi popravki nastavitev, tudi pri Logitech kameri dosegli podoben rezultat. Tako smo pri razvoju sistema uporabili kamero EKACOM, saj je cenovno ugodnejša, pri isti uspešnosti kot dražja alternativa.

Nazadnje smo preverili, če postavitev kamere na različnih lokacijah okoli tarče vpliva na uspešnost. Izbrali smo postavitev na levi, desni in zgornji strani tarče. Na vse lokacije smo postavili identične kamere, pod istim kotom glede na tarčo. Ugotovili smo, da lokacija postavitve kamere ne vpliva na uspešnost zaznavanja. Na vseh lokacijah se pojavlja problem prekrivanja, njihovo število je naključno, odvisno od pozicije puščic. Dosežena uspešnost na poziciji desno in levo od tarče je bila slabih 97 %, iz zgornje strani pa 96%.

Edini problem, ki je ostal po vseh prilagoditvah je prekrivanje puščic. Enostavno in učinkovito rešitev za to težavo, smo dobili pri testiranju zadnje prilagoditve. Čeprav se je na vsaki izmed lokacij pojavljalo prekrivanje, smo ugotovili, da problemi niso bili prisotni na istih primerih. Izkazalo se je, da sta v vsaki situaciji, vsaj dve kameri vrnili točno vrednost. Eden izmed razlogov so različne lokacije kamer. Ker je razmak med njimi približno 120°, obstaja zelo majhna možnost, da bi bila ena puščica zakrita iz dveh lokacij. V primeru, ko vsaj dve kameri vrneta isto vrednost, privzamemo, da je to pravilen rezultat. S tem pristopom smo na isti testni množici dosegli uspešnost 100 %. Slika 4 prikazuje, končno konfiguracijo sistema s tremi kamerami.

Slika 4: Konfiguracija sistema po vseh prilagoditvah.
Vir: lasten.

4 Zaključek

V članku smo opisali razvoj in potrebne prilagoditve sistema za detekcijo puščic pri klasičnem pikadu. Ugotovili smo, da zaradi težav s prekrivanjem ena kamera ne zadošča za doseganje visoke uspešnosti. V končni različici sistema smo uporabili tri kamere in na testni množici dosegli 100 % uspešnost.

V prihodnje načrtujemo izdelavo sistema, še z uporabo metod računalniškega vida, ki temeljijo na strojnem učenju. Menimo, da bi s tem pristopom dobili bolj robusten sistem, ki bi bil bolj prilagodljiv na spremembe okolice.

Literatura

Laganiere, R. (2017). OpenCV 3 Computer Vision Application Programming Cookbook, tretja izdaja. Packt Publishing Limited.

Scolia. (2023) https://scoliadarts.com/.

Chhikara, P. (2022). Understanding morphological image processing and Its operations. https://towardsdatascience.com/understanding-morphological-image-processing-and-its-operations-7bcf1ed11756.

Soille, P. (1999). Morphological Image Analysis, Principles and Applications. (Vol. 2, No. 3, pp. 170-171). Berlin: Springer.