Optični tok s postopkom Horn-Schunck

Matej Tomc

Univerza v Ljubljani, Fakulteta za elektrotehniko, Tržaška 25, 1000 Ljubljana, Slovenija
E-pošta:mt8907@student.fe.uni-lj.si

Povzetek. V poročilu je predstavljeno teoretično ozadje izračuna optičnega toka po metodi Horn-Schunck. Natančno je predstavljena klasična metoda algoritma, skupaj z njenimi praktičnimi omejitvami. Predstavljena je tudi metoda z uporabo piramidne strukture, ki omogoča izračun optičnega toka pri večjih odmikih. V prilogi je podana koda v jeziku Python, ki implementira oba algoritma ter implementacija enostavne aplikacije stabilizacije posnetka na podlagi optičnega toka.

Ključne besede: optični tok, metoda Horn - Schunck, stabilizacija posnetka

Horn-Schunck method for calculating optical flow

Abstract. This report covers the theoretical background of optical flow calculation using Horn-Schunck method. The classical method is discussed in detail, including its limitations. A better approach using a pyramidal scheme that allows for accurate calculation even when larger displacements are present is discussed briefly. A Python implementation of both algorithms is provided in the appendix as well as a simple application for video stabilization that is based on optical flow calculation.

Keywords: optical flow, Horn- Schunck method, video stabilization

1 UVOD

Ena sama slika nam lahko pove nekaj o razdaljah med objekti na sliki, vendar ne nosi dovolj informacije, da bi lahko povedala kaj o gibanju objektov na sliki. Za oceno gibanja je potrebno zaporedje vsaj dveh slik. Tej oceni gibanja pravimo optični tok. Optični tok je porazdelitev navideznih hitrosti gibanja vzorcev svetlosti na sliki. Navadno se pojavi zaradi relativnega gibanja med opazovalcem in objektom na sliki. [1]

Podatek o optičnem toku navadno za vsako točko v sliki predstavimo z vektorjem gibanja $\boldsymbol{h} = (u, v)^T$, kjer u predstavlja premik v smeri x in v premik v smeri y pri dveh zaporednih slikah in.

Za oceno optičnega toka poznamo več algoritmov, ki delujejo na različnih principih. Glavni uveljavljeni principi ocenjevanja optičnega toka izhajajo iz izračuna diferencialov, ujemanja regij, izračuna energije ali izračuna faze.[2]. V članku se bom omejil na obravnavo in implementacijo ocene optičnega toka po metodi, ki sta jo leta 1980 razvila Berthold K.P. Horn in Brian G. Schunck.[1]. Postopek Horn-Schunk deluje na principu izračunavanja diferencialov.

2 ALGORITEM HORN-SCHUNCK

2.1 Predpostavka o konstanti svetlosti

Naj oznaka $I(x, y, t)$ predstavlja zaporedje slik. Naj bo $\big(x(t), y(t)\big)$ trajektorija neke točke preslikana v ravnino slike. Predpostavka o konstantni svetlosti trdi, da so svetlosti pikslov,$I(x(t), y(t), t)$ konstante po času:

$$\frac{dI}{dt} = 0 \tag{1}$$

Ta predpostavka naj velja za vsako točko na sliki. Upoštevaje verižno pravilo odvajanja dobimo enačbo $\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$, ki jo lahko ob upoštevanju našega dogovora za označevanje vektorja gibanja $\boldsymbol{h} = (u, v)^T = (\frac{dx}{dt}, \frac{dy}{dt})^T$ zapišemo kot

$$\nabla I * \boldsymbol{h} + I_t = 0 \tag{2}$$

, pri čemer je $\nabla I = \left(\frac{dI}{dx}, \frac{dI}{dy}\right) = \big(I_x, I_y\big)$ in $I_t = \frac{dI}{dt}$. Enačba (2) je **enačba optičnega toka**. Ta enačba pa nam ne zadostuje za dokončno določitev optičnega toka, saj v vsaki točki iščemo vrednost dveh neznank u in v, imamo pa samo eno enačbo.

Razmislimo, zakaj je predpostavka o konstantni intenziteti smiselna, s pomočjo primera. Denimo da kamera snema streho vlaka. Denimo da je vlak rdeče kovinske barve. Če na vlak sije sonce, na strehi zaradi odboja pride do tega, da je na določenem mestu slike streha svetlejša, na drugem koncu pa recimo zakrita s senco in temnejša. Če se vlak premakne sta odboj sonca in senca še vedno na istem mestu in tako iz slikovne informacije ne moremo dobiti informacije o gibanju vlaka. Če bi svetla lisa ne bila posledica sončne svetlobe in temni del ne posledica sence, ampak bi bila to drugačna barva na strehi vlaka, bi se premaknila skupaj z

Prejet d. mesec, leto
Odobren d. mesec, leto

vlakom in tako bi lahko dobili podatek o hitrosti gibanja vlaka iz posnetka.

Naša predpostavka o konstantni intenziteti točk torej v praksi pomeni, da moramo zagotoviti osvetlitev, ki ne meče senc in ne dodaja popačenj zaradi odbojev. Prav tako zaradi postopka izračuna z odvodi ni dobro, da so v sliki veliki skoki v intenziteti.

2.2 Predpostavka o gladkosti

Zaenkrat imamo s postavitvijo omejitve na konstantnost pikslov naš prostor rešitev za *u* in *v* zmanjšan iz ravnine v premice, a še vedno ne vemo, katera izmed neskončno rešitev na premici je prava vrednost optičnega toka. Pri izbiri druge omejitve imamo več možnosti izmed katerih sta Horn in Schunck predlagala predpostavko o gladkosti. Ta predpostavka veli, da mora biti optični tok izračunan v določeni točki podoben optičnemu toku v točkah v njeni okolici. Posledično morajo za pravilni izračun optičnega toka biti premiki v sliki primerno majhni. Poglejmo si to predpostavko še v bolj formalnem zapisu:

$$\left(\frac{du}{dx}\right)^2 + \left(\frac{dv}{dx}\right)^2 + \left(\frac{du}{dy}\right)^2 + \left(\frac{dv}{dy}\right)^2 \qquad (3)$$

Če gornji izraz (3) minimiziramo, bomo poskrbeli, da so med bližnjimi točkami razlike v optičnem toku majhne.

Osmislimo si pomen te predpostavke. Denimo da spet gledamo s kamero železniško postajo. Z vrha vidimo streho rdečega vlaka, v ozadju slike pa so preostale proge. V primeru da se vlak premakne, se nam zdi logično, da bodo vektorji gibanja kazali v isto smer in bodo enako veliki po celotni strehi premikajočega se vlaka. Toda brez predpostavke o gladkosti temu ne bi nujno bilo tako. Katerikoli rdeči piksel, ki predstavlja streho vlaka, bi se lahko preslikal v nek drugi rdeči piksel na naslednji sliki posnetka, pa bi bil pogoj o konstantni svetlosti še vedno izpolnjen. Šele predpostavka o gladkosti nam da smiselne rezultate pri premiku velikega togega telesa na sliki.

Izjema se pojavi tedaj, ko hočemo optični tok uporabiti za razpoznavo predmeta v ospredju iz ozadja. Tedaj si želimo da se v optičnem toku na robu predmeta pojavijo nezveznosti, saj na ta način lahko identificiramo predmet.

2.3 Metoda Horn-Schunck kot variacijski problem

Sedaj ko poznamo obe predpostavki, ju združimo v skupno cenilko, minimizacija katere nam bo dala rešitev za *u* in *v*.

$$J(\boldsymbol{h}) = \int_{\Omega} \left(I_x u + I_y v + I_t\right)^2 + \alpha^2(|\nabla u|^2 + |\nabla v|^2) \qquad (4)$$

Enačba (4) je torej variacijski problem, kjer iščemo minimum funkcionala $J(\boldsymbol{h})$. Integral po omega predstavlja integracijo po celotni sliki. Če bi enačbo pisali v diskretnem prostoru, bi integral nadomestili z vsoto po vseh (x,y), to je po vseh točkah slike. Prvi izraz v integralu prepoznamo kot enačbo optičnega toka, ki nosi v sebi predpostavko o konstantni svetlosti. Drugi izraz predstavlja predpostavko o glasnosti, kjer smo enačbo (3) zapisali na krajši način in jo pomnožili z α^2. Parameter alfa je v enačbi (4) prosti parameter, ki med seboj uteži vpliv prvega in drugega izraza na skupno vrednost funkcionala. Literatura priporoča, da $\alpha^2 < 1$, saj s tem poskrbimo, da ima enačba optičnega toka še vedno večji vpliv na rezultat kot pogoj o gladkosti. [5]

Sedaj ko poznamo cenilko, se lahko lotimo optimizacije. Pri optimizaciji računamo, da bomo dosegli lokalni minimum naše cenilke. Minimizacija funkcionala (4) nam rezultira v seldečih Euler-Lagrangevih enačbah:

$$I_x^2 u + I_x I_y v = \alpha^2 \mathrm{div}(\nabla u) - I_x I_t \qquad (5)$$

$$I_x I_y u + I_y^2 v = \alpha^2 \mathrm{div}(\nabla v) - I_y I_t \qquad (6)$$

Enačbi si poenostavimo s približkom za Laplaceov operator:

$$\mathrm{div}(\nabla u) \approx \bar{u} - u \qquad (7)$$

kjer enako velja tudi za *v*. $\bar{u}$ v enačbi predstavlja lokalno povprečje optičnega toka v smeri *x*.

Če v sistem enačb (5) in (6) vstavimo približek (7) in premečemo izraze dobimo sledeči sistem dveh enačb:

$$u = \bar{u} - \frac{I_x\left(I_x\bar{u} + I_y\bar{v} + I_t\right)}{(\alpha^2 + I_x^2 + I_y^2)} \qquad (8)$$

$$v = \bar{v} - \frac{I_y\left(I_x\bar{u} + I_y\bar{v} + I_t\right)}{(\alpha^2 + I_x^2 + I_y^2)} \qquad (9)$$

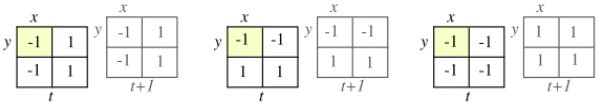
2.4 Algoritem

Enačbi (8) in (9) dajeta sistem, ki ga lahko enostavno rešimo z iterativno metodo. V vsaki iteraciji *u* in *v* na levi izračunamo z vrednostmi iz prejšnje iteracije, ki jih uporabimo za izračun izraza na desni. Kot začetni približek optičnega toka v vsaki točki, sta Horn in Schunck predlagala $(u, v) = (0,0)$ za vsako točko v sliki, to ni edini in verjetno ne najboljši možen začetni približek za vsak par slik, a iteracija nas pripelje do prave rešitve.

Poglejmo si sedaj, kako izračunamo posamezne spremenljivke v enačbah (8) in (9).

Vrednost lokalnega povprečja $\bar{u}$ lahko ocenimo kot eno četrtino vsote optičnega toka piksov nad, pod, levo in desno od piksla za katerega trenutno računamo povprečje. Možnost, ki sem jo uporabil v implementaciji pa upošteva še diagonalne sosede. Diagonalne sosede uteži z 1/12, ostale pa z 1/6. Ker $\bar{u}$ nastopa na desni strani enačbe, moramo biti pozorni, da za izračun povprečja uporabimo vrednosti prejšnje iteracije za vsak piksel in v isti matriki nikoli ne hranimo del pikslov iz trenutne in del pikslov iz prejšnje iteracije.

Odvodi I_x, I_y in I_t so znotraj posameznih iteracij konstante. Odvodi vsebujejo podatek o dveh slikah, ob času t in t+1, zato za izračun parcialnih odvodov uporabljamo konvolucijska jedra, predlagana s strani Horna in Schuncka, ki so prikazana na sliki (1)



Slika 1: Konvolucijska jedra za izračun parcialnih odvodov.

2.5 Metoda Horn-Schunck s hierarhično strukturo

2.5.1 Slabosti klasičnega algoritma

Do sedaj sem v članku govoril le o klasičnem algoritmu Horn-Schunck. Težava tega algoritma je, da lahko zaradi svojih omejitev optični tok dobro oceni le za zelo majhne premike, v praksi reda velikosti en piksel. Omejitev izhaja predvsem iz naslova ocene parcialnih odvodov, katerih približek dobro poznamo le na zelo majhnem območju. Spomnimo se namreč, da je jedro za izračun približkov parcialnih odvodov dimenzij 2x2x2 in je ta kocka osmih pikslov edino, kar upoštevamo.

Ta omejitev je za praktično uporabnost algoritma povsem prehuda, saj ni realno pričakovati, da bo gibanje objektov na sliki omejeno na en piksel. Za oceno večjih premikov obstaja modifikacija algoritma, ki s piramidno strukturo dobro oceni tudi večje premike.

2.5.2 Algoritem s piramidno strukturo

Zamenjajmo enačbo (2) s sledečo enačbo:

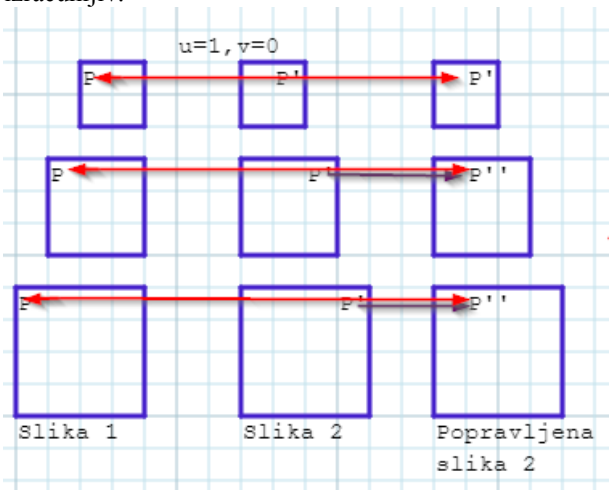
$$I(x,y,t) - I(x+u, y+v, t+dt) = 0 \qquad (10)$$

Enačba (10) še vedno opisuje predpostavko o konstantni svetlosti, če vzamemo da sta sliki ob času t in ob $t+dt$ dve zaporedni sliki. Razlika med zaporednima slikama torej nadomešča vpliv časovnega odvoda. Razlika od klasičnega algoritma je v tem, da se sedaj svetlost ohranja, toda odmik med točkama iste svetlosti (u,v) je lahko poljubno velik. Predpostavka o gladkosti ostane enaka. Novi funkcional, ki ga bomo minimizirali se sedaj glasi:

$$J(\boldsymbol{h}) = \int_{\Omega} \left(I(x,y,t) - I(x+u, y+v, t+dt)\right)^2 + \alpha^2\left(|\nabla u|^2 + |\nabla v|^2\right) \qquad (11)$$

Polnega postopka na tem mestu ne bom razlagal, saj je v celoti povzet iz literature [3]. V grobem minimizacija poteka tako, da iz funkcionala spet dobimo sistem dveh Euler-Lagrangevih enačb, ki jih linearizirano s pomočjo razvoja v Taylorjevo vrsto.

Predlagani algoritem s piramidno strukturo sloni na sledečem razmisleku: Spoznali smo, da je mogoče dobro oceniti optični tok, če so premiki v sliki dovolj majhni. Če sliko decimiramo na več nivojih v piramidno shemo, se bodo tudi absolutne razdalje na sliki zmanjšale z višanjem nivoja v piramidi. Na ta način lahko z zadostno decimacijo slike pridemo na tako majhne sličice, da bodo vsi premiki majhni in bomo optični tok za ta nivo decimacije lahko dobro ocenili. Ko imamo na določenem nivoju izračunan optični tok, se lahko spustimo na nivo nižje (,kjer je slika višje resolucije), pri tem pa informacijo iz prej izračunanega nivoja uporabimo tako, da piksle premaknemo tako, kakor nam veli izračunani optični tok. Če smo bili pri oceni optičnega toka uspešni, bo ta premik pomenil, da je originalna slika na trenutnem nivoju sedaj bolj podobna popravljeni drugi sliki na trenutnem nivoju, kot pa bi bila podobna originalni drugi sliki, ki je nismo premaknili. Tako upamo, da bo tudi na trenutnem nivoju med prvo in popravljeno drugo sliko optični tok majhen in ga bomo lahko spet pravilno ocenili. Na ta način se po piramidi pomikamo proti najnižjemu nivoju, to je originalnemu paru slik, in s sprotnim popravljanjem slike skrbimo, da optični tok v vsakem računskem koraku ostane majhen in tako dobro izračunljiv.



Slika 2: Prikaz delovanja algoritma s piramidno strukturo. Namesto da bi med seboj primerjali sliko 1 in sliko 2, na vsakem nivoju primerjamo sliko 1 s popravljeno sliko 2 in izračunan optični to kot popravek vnašamo na nižji nivo. Optični tok tako ostaja majhen.

3 MOŽNOSTI APLIKACIJ

Podatek o optičnem toku lahko s pridom uporabimo v različnih aplikacijah. Aplikacija, ki sem jo implementiral je namenjena stabilizaciji slike. Relativno gibanje namreč ni nujno le posledica gibanja objektov, ampak gre lahko tudi za gibanje roke ki drži kamero. Če večino slike predstavlja ozadje, lahko ugotovimo kakšen je navidezni premik celotne slike in to kompenziramo.

V zadnjem času zelo popularna aplikacija je algoritem za izogibanje oviram uporabljen na kvadrikopterjih. Pri premiku drona se objekti, ki so mu blizu nesorazmerno bolj premaknejo v sliki kot tisti oddaljeni. Iz te nezveznosti je mogoče sklepati katere ovire bodo dronu na poti in katere so oddaljene. Ker optični tok daje tudi aproksimacijo gibanja, lahko dron tudi napove, če mu kateri drugi objekt v vidnem polju namerava zapreti pot in se ustrezno odmakne.

Na istem principu deluje še razpoznava oblik objektov v ospredju, optični tok pa se lahko uporablja tudi za višanje ločljivosti videa pri interpolaciji, sledenje vozilom v prometi, zaznavo gest v uporabniškem vmesniku in še bi lahko naštevali.

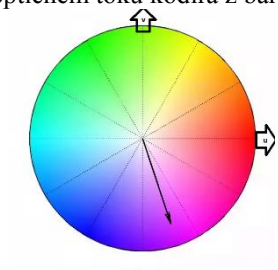
4 IMPLEMENTACIJA

Cilj moje aplikacije je bil stabilizirati video zajet s kamero, ki sem jo držal v roki.

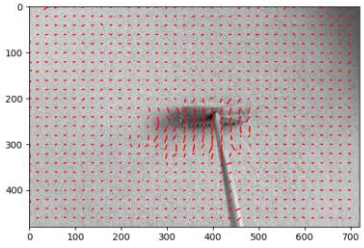
Obe različici algoritma sem implementiral v programskem jeziku Python. Koda je dostopna na naslovu https://github.com/traparije/koncniRV. Kodi so dodani ustrezni komentarji. Pri implementaciji sem čim tesneje sledil teoriji.

Edino mesto, kjer sem pri implementaciji odstopil od teorije, je v notranji zanki, ki numerično rešuje optični tok za vsako popravljeno sliko. Literatura [3] predlaga uporabo SOR metode (ang. Successive Over-Relaxation), ki pa mi ni skonvergirala. Med računanjem je namreč prišlo do numeričnih napak. Namesto tega sem uporabil Gauss-Seidlovo metodo, ki je pravzaprav poseben primer SOR metode [4] in z njo sem dosegel dobre rezultate.

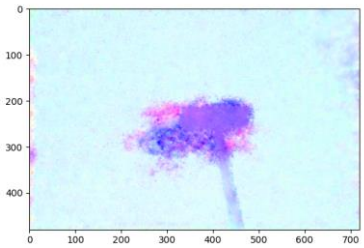
Za vizualizacijo optičnega toka sem pripravil dve funkciji. Prva na sliko doriše polje vektorjev, druga pa informacijo o optičnem toku kodira z barvo.



Slika 3: Optični tok lahko prikažem z risanjem vektorja ali z barvo. Slika kaže kako sem barvno kodiral optični tok



Slika 4: Optični tok prikazan z vektorji kot posledica premika kamere v levo stran in metle navzdol.



Slika 5: Optični tok prikazan z barvo kot posledica premika kamere v levo stran in metle navzdol.

Za demonstracijo delovanja poravnave si v mapi https://github.com/traparije/koncniRV/tree/master/Rezultati poglejte datoteki out_neobdelan_gray.mp4 in out_obdelan.avi in jih med seboj primerjajte.

5 Ocena delovanja

Če kritično precenimo vhodni in poravnani posnetek, opazimo, da posnetek vedno slabše poravnava sceno. Eden izmed razlogov za to je, da za popravek ves čas gledam le mediano optičnega toka v prepričanju da je večina slike ozadje, katerega optični tok bo le posledica premika kamere. Žal pa s tem poskrbim le za translacijsko korekcijo medtem ko rotacije ne korigiram. Prav tako ne korigiram morebitnega bližanja ali oddaljevanja kamere.

Druga težava je, da vse slike poskušam poravnati na sceno prve slike v posnetku, na voljo pa imam le izračun med posameznimi zaporednimi slikami. Tako korekcijske parametre ves čas le seštevam in napaka se lahko samo kopiči.

Za prikaz popolne poravnave imam sicer naložene v mapi https://github.com/traparije/koncniRV/tree/master/Frames slike iz posnetka statične scene odseva laboratorija v kovinski steni, a taka popolna poravnava ne omogoča gibanja predmetov v posnetku.

6 Zaključek

Algoritem je v obeh verzijah pravilno implementiran. Klasični algoritem deluje na res zelo majhnih odmikih, medtem ko sem z algoritmom s piramidno strukturo uspešno določeval optični tok tudi pri odmikih okrog 15 pikslov. Implementacija v Pythonu je relativno hitra, a vseeno ne dovolj učinkovita, da bi upal računati na dovolj natančne rezultate v realnem času. Verjetno bi uspešnost algoritma lahko še precej izboljšal, če bi izbral bolj primerne proste parametre.

Literatura

[1] Horn B.K.P in Schunck B.G.: Determining Optical Flow, Artificial inteligence pp 185-203, (1980) http://image.diku.dk/imagecanon/material/HornSchunckOptical_Flow.pdf #HS original članek

[2] Barron J.L., Fleet D.J. in Beauchemin S.S.: Performance of optical Flow Techniques, IJCV 12:1, pp.43-77 (1994), http://www.cs.toronto.edu/~fleet/research/Papers/ijcv-94.pdf

[3] Enric Meinhardt-Llopis, Javier Sánchez Pérez, in Daniel Kondermann: Horn-Schunck Optical Flow with a Multi-Scale Strategy,Image Processing On Line, 3 (2013), pp. 151–172. https://doi.org/10.5201/ipol.2013.20

[4] Plestenjak Bor: Numerične metode, (2010), https://www.fmf.uni-lj.si/~plestenjak/Vaje/NaFgg/Predavanja/OldKnjiga_NM.pdf?fbclid=IwAR1Q_PcwUNaEsJYP3EfglmCoX2CCVREytmadE48uIpWRKFCLnysyLyk8YuM

[5] Klette R.: Concise Computer Vision - An Introduction into Theory and Algorithms. Springer London (2014)

Priloga

Vsa koda in uporabljene testne slike in posnetki so dostopni na: https://github.com/traparije/koncniRV.