

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Elementarni postupak diferencijalnog praćenja **Tehnička dokumentacija** **Verzija 1.0**

Studentski tim: Kristijan Biščanić
Matej Djerđji
Ivan Fabijanić
Josip Milić
Dario Pažin

Nastavnik: Prof. dr.sc. Siniša Šegvić

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Sadržaj

1. Uvod	3
2. Početna obrada slike	5
2.1 Isticanje rubova uz pomoć Sobelovog operatora	6
2.2 Zaglađivanje rubova Gausovim filterom	7
2.3 Programska implementacija početne obrade slike	9
3. Harrisovi kutovi kao značajke slike	10
3.1 Harrisovi kutovi	10
3.2 Programska implementacija detektora Harrisovih kutova	12
4. Algoritam KLT pratitelja	14
5. Programska implementacija algoritma KLT pratitelja	17
5.1 Praćenje značajki između dvije slike	17
5.2 Praćenje značajki u video datoteci	19
6. Eksperimentalni rezultati i problemi	21
7. Zaključak	24
8. Upute za korištenje programske implementacije	25
8.1 Prototip: Upute za korištenje	25
8.2 Završni program: Upute za korištenje	26
9. Literatura	28

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

1. Uvod

Ljudski vid je osjetilo pomoću kojeg čovjek ostvaruje percepciju objekata ili pojava iz njegove okoline. Područje računalnog vida je područje umjetne inteligencije koje istražuje navedenu percepciju u svrhu nalaženja načina za što vjernije računalno oponašanje ljudskog vida i rješenja koja služe kao pomoć čovjeku u razumijevanju njegove (ne)posredne okoline. Umjesto korištenja ljudskog oka i mozga uz biokemijske procese za ostvarenje vida, računalno koristi vanjsku jedinicu sa sposobnošću pretvorbe svjetlosnih signala u digitalni oblik i procesne jedinice koje složenim matematičkim operacijama obrađuju dobivene podatke. Od mnogih problemskih zadataka koji su iz domene područja računalnog vida najčešći su prepoznavanje objekata, analiza pokreta, rekonstrukcija i restauracija slika. Neki od primjera tih zadataka: prepoznavanje lica, praćenje značajki u video snimci, stvaranje 3D modela temeljem slika i video snimke, čišćenje šuma na slikama kao pomoć u dijagnostici i mnogi drugi. Rješenja većine tih problema postoje već desetljećima, ali tek snažnim razvojem računala (algoritmi rješenja, ukoliko se želi prihvatljivo brzo dobiti koristan rezultat, nerijetko zahtijevaju velike računalne performanse) u novije doba dolaze do izražaja upotrebom u mnogim granama ljudske djelatnosti (medicina, poljoprivreda, filmska umjetnost i mnoge druge). Daljnjim razvojem i povećavanjem računalnih performansi, rješenja tih problema (i ona koja će tek nastati) će se sigurno sve više implementirati u mnoge postojeće djelatnosti čineći ih efikasnijim i(li) kvalitetnijim. Poučeni računalnom poviješću možemo pretpostaviti i stvaranje djelatnosti koje su usko vezane uz područje računalnog vida, a danas su nam gotovo nezamislive.

Elementarni postupak diferencijalnog praćenja je problemski zadatak analize pokreta u kojem se traži praćenje određenih značajki u nizu slika. Jedno od mogućih rješenja tog problema je Kanade-Lucas-Tomasi pratitelj značajki¹ (skraćeno KLT pratitelj) i njegov se algoritam koristi i obrađuje u ovom projektu. Značajka je ime za značajnu informaciju koja je relevantna daljnjim operacijama kojima se dolazi do konačnog rješenja. U slučaju ovog projekta, značajke su Harrisovi kutovi koji se nađu na određenim slikama u nizu slika. Zbog ubrzanja i preciznosti koristi se obrada slike na način da se rubovi istaknu uz pomoć Sobelovog operatora i izgade pomoću Gaussovog filtera. KLT algoritam, pojednostavljeno opisan, čita podatke slike, obrađuje ih za lakše i brže čitanje, pronalazi značajke (Harrisove kutove), u sljedećim slikama pronalazi nove položaje tih značajki pomoću određenih metoda i iterativno ih uspoređuje s prethodno nađenim položajima značajki. Prikazuje one značajke kojima je pronašao novi položaj, a ostale odbacuje. Proces ponavlja za svaki podniz niza slika sve dok ne završi s cijelim nizom slika. Rezultat tog postupka odnosno projekta je niz slika s označenim značajkama u kojem se može jasno vidjeti praćenje i gubljenje nađenih značajki.

¹ Engleski naziv: *Kanade-Lucas-Tomasi feature tracker*

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Programsko ostvarenje projekta sastoji se od prototipa i završnog programa. Prototip, implementiran Pythonom (v2.7), skriptnim jezikom, ima funkciju prikaza Harrisovih kutova na pojedinoj obrađenoj slici. Njegovim izvođenjem se može vidjeti koliko na brzinu izvedbe utječe odabir pojedinog programskog jezika odnosno kako je izvedba u C++ jeziku mnogo brža od one u Python jeziku. Prototipom se, u ranoj fazi projekta, predviđela zahtjevnost ostatka projekta i temeljem toga se obratila potpuna pozornost na izvedbu u C++ jeziku. Završni program, implementiran s C++-om, jezikom opće namjene, koristi algoritam KLT pratitelja uz pomoć kojeg ostvaruje praćenje značajki u nizu slika. Prototip i završni program imaju vrlo jednostavno GUI sučelje uz pomoć kojeg se prikazuje slika s nađenim značajkama u slučaju prototipa odnosno niz slika s nađenim značajkama (koje se prikazuje u stvarnom vremenu prilikom postupka) u slučaju završnog programa. Značajka se na (pojedinoj) slici označava sitnim crvenim rombom.

Ulaz prototipa je slikovna datoteka, a izlaz je kopija ulazne datoteke s označenim značajkama koja se može otvoriti neovisno o prototipu. Ulaz završnog programa može biti video datoteka ili niz pojedinih slikovnih datoteka. Izlaz završnog programa je kopija video datoteke ili niza pojedinih slikovnih datoteka s označenim značajkama koja se može pokrenuti neovisno o završnom programu.

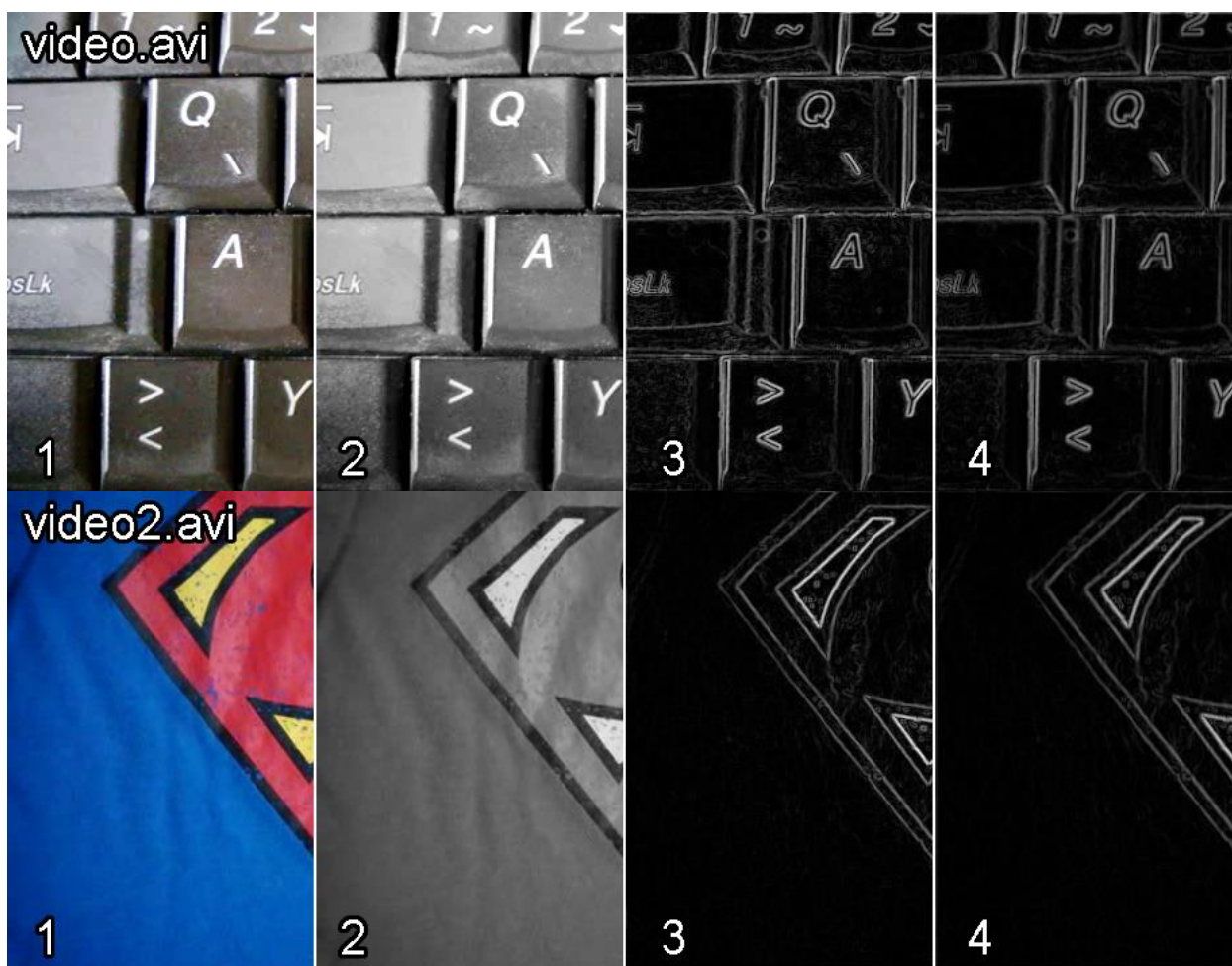
U svrhu prevođenja i testiranja programskog koda za prototip se koristio standardni Python IDLE, a za završni program Microsoft Visual Studio 2010. Obje verzije programa su koristile biblioteku programskih funkcija OpenCV (v2.4.6) koja je omogućila čitanje i ispis slikovnih odnosno video datoteka. Treba napomenuti instalaciju biblioteke OpenCV kao nužnost kako bi se mogao prevesti programski kod i složenost instalacije te biblioteke u C++ okruženju, točnije Microsoft Visual Studio okruženju što je, objašnjeno u 8. poglavlju (Upute za korištenje) koje bi se trebalo pročitati prije vlastitog prevođenja i testiranja programskog koda.

U sljedećim poglavljima bit će detaljno prikazano sve navedeno: obrada slika (2. poglavlje), traženje značajki (Harrisovih kutova) na slici (3. poglavlje), općeniti algoritam KLT pratitelja (4. poglavlje), programska implementacija navedenog algoritma (5. poglavlje), eksperimentalni rezultati (6. poglavlje), zaključak (7. poglavlje), upute za korištenje (8. poglavlje) i popisa literature (9. poglavlje).

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

2. Početna obrada slike

Prije same uporabe KLT algoritma u postupku, korisno je sliku obraditi. U ovom projektu se koristi Gaussov filter za zaglađivanje slike čime se smanjuje šum (nekorisna informacija) i Sobelov operator za isticanje rubova slike što pojednostavljuje pronalazak Harrisovih kutova, značajki (značajnih informacija slika) koje se koriste u ovom projektu. Prije obrade, slika se prebacuje u crno-bijelu verziju slike čime se pojednostavljuje postupak obrade (umjesto tri RGB² vrijednosti koje se koriste za prikaz određene boje piksela u originalnoj verziji koristi se samo jedna vrijednost nijansi sive u obrađenoj verziji). Slika 2 prikazuje korake obrade na dva primjera.



Slika 2 - Originalna slika (1) se prebacuje u crno-bijelu verziju (2), ističe Sobelovim operatorom (3) i zaglađuje Gaussovom filtrom (4).

² RGB je akronim za aditivni model boja u kojem se određena boja piksela prikazuje kombinacijom nijansi crvene (Red), zelene (Green) i plave (Blue) boje. Vrijednosti nijansi mogu biti od 0 do 255.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

2.1. Isticanje rubova uz pomoć Sobelovog operatora

Za pronalazak Harrisovih kutova na slici potrebno je izračunati gradijent slike. To je moguće napraviti primjenom Sobelovog operatora koji daje dobru aproksimaciju gradijenta slike. Sobelov operator sastoji se od dva kernela (kvadratne matrice) – jedan za izračun derivacije po x – smjeru i drugi za izračun derivacije po y – smjeru. Kombinacijom rezultata primjene pojedinih kernela na sliku za koju treba izračunati gradijent, moguće je izračunati gradijent prema:

$$G = \sqrt{G_x^2 + G_y^2}, \quad \theta = \tan^{-1} \frac{G_y}{G_x}$$

gdje je G amplituda gradijenta, θ kut gradijenta, a G_x i G_y su derivacije po x i y smjeru dobivene primjenom Sobelovih kernela. Često se zbog povećanja performansi uvodi aproksimacija za izračun amplitude:

$$G = |G_x| + |G_y|.$$

Sobelovi kerneli definirani su prema (n1). Ovakav format kernela implicira smjer pozitivne x-osi „s lijeva na desno“, a y-osi „gore prema dolje“.

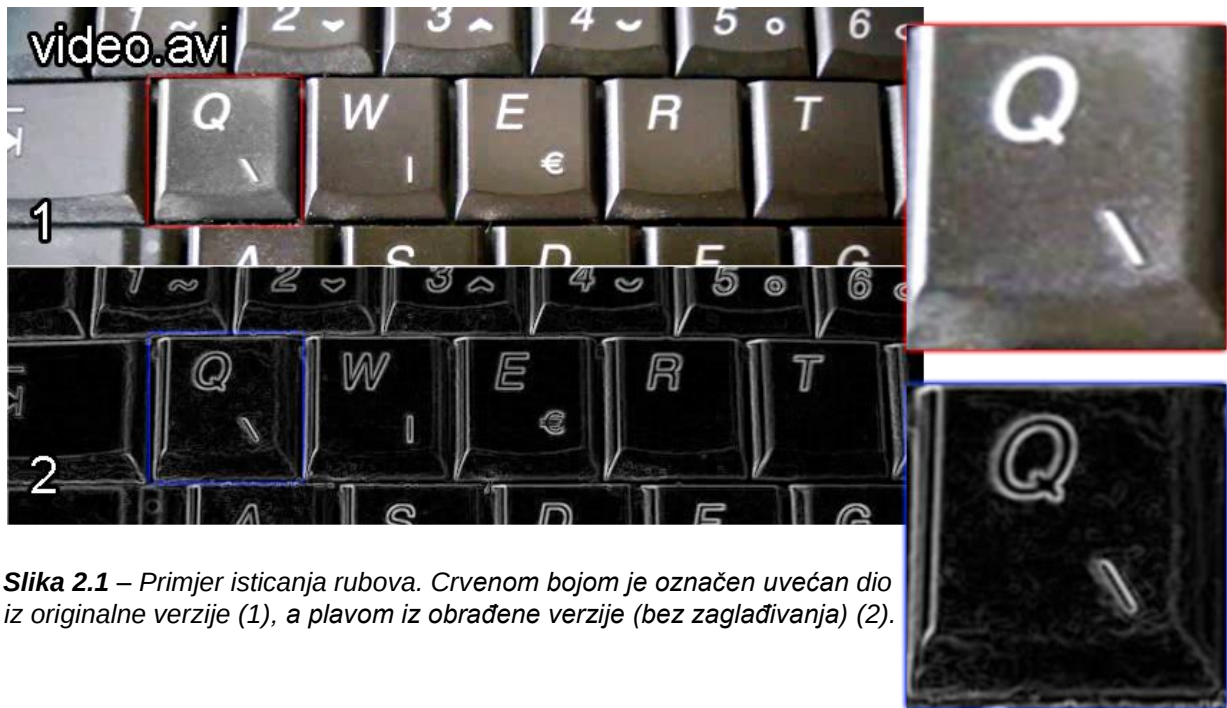
$$(n1) \quad G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix}, \quad G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Kerneli se primjenjuju tako da se svaki posebno konvoluiraju sa izvorišnom slikom, a pošto su i ovi kerneli separabilni, broj potrebnih operacija za računanje derivacije u jednom smjeru za svaki piksel smanjuje se sa k^2 na $2 \cdot k$, gdje je k veličina kernela (u ovom slučaju $k=3$). Separirani kerneli prikazani su u (n2).

$$(n2) \quad G_x = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} +1 & 0 & -1 \end{bmatrix}, \quad G_y = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Slika 2.1 (sljedeća stranica) prikazuje isticanje rubova na jednom primjeru. Posebno su označeni uvećani dijelovi koji služe za bolju preglednost.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13



Slika 2.1 – Primjer isticanja rubova. Crvenom bojom je označen uvećan dio iz originalne verzije (1), a plavom iz obrađene verzije (bez zaglađivanja) (2).

2.2. Zaglađivanje rubova Gaussovim filtrom

Zbog nesavršenosti kamera i fotoaparata zapisane digitalne fotografije sadrže određenu količinu Gaussovog šuma. Gaussov šum je signal čije vrijednosti podliježu Gaussovoj (normalnoj razdiobi), te su neovisne o poziciji i intenzitetu piksela na koji se pribrajaju. Kako bi se u krajnjim rezultatima smanjio utjecaj šuma, koji ne nosi nikakvu korisnu informaciju, potrebno je sliku prije bilo kakve daljnje obrade zagladiti. Kod zaglađivanja slike koristi se Gaussov filter, $G(x, y)$:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}.$$

U funkciji $G(x, y)$ veličina σ naziva se standardna devijacija i u ovom slučaju određuje mjeru „glađenja“: korištenjem manje veličine σ moguće je da se neće ukloniti sav šum, dok je korištenjem prevelikog σ moguće ukloniti i detalje slike koji su ustvari korisni kod daljnje obrade.

Gaussov filter primjenjuje se tako da se funkcija $G(x, y)$ konvoluirana sa zadanom slikom $I(x, y)$. U ovom projektu se koriste diskretne slike i zato se koristi diskretan Gaussov filter i ostvaruje se kernelom - kvadratnom matricom dimenzije $2k+1$. Vrijednosti kernela H su iznosi kontinuirane funkcije $G(x, y)$ na određenim koordinatama:

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

$$H(i, j) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-k-1)^2 + (j-k-1)^2}{2\sigma^2}\right), \quad i, j \in (-k, k).$$

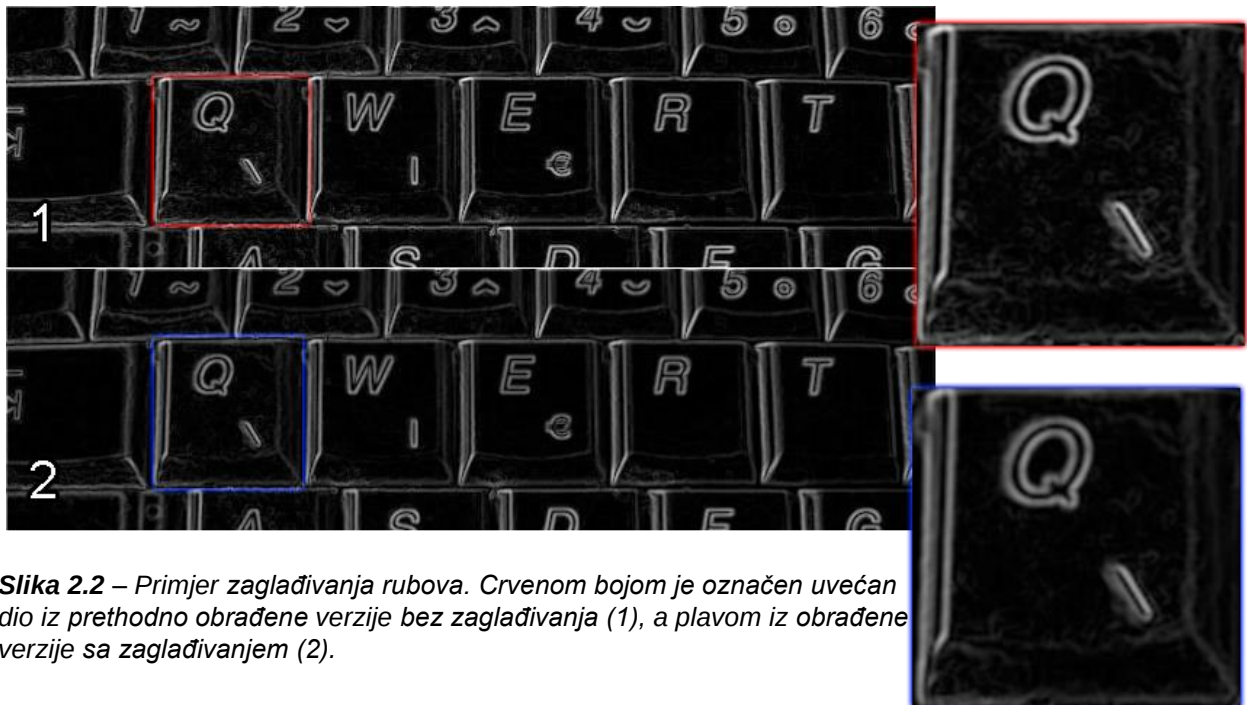
Gaussov filter moguće je rastaviti na x i y komponentu, te primijeniti svaku komponentu zasebno. Time se broj operacija potrebnih za računanje vrijednosti svakog piksela smanjuje sa k^2 na $2k$. Potrebno je kernel H rastaviti na jedan vektor-redak h , koji pomnožen sa istim takvim, ali transponiranim retkom h^T daje polazni kernel H . Vrijednosti vektora h dimenzije k računaju se pomoću jednodimenzionalne Gaussove funkcije:

$$h(i) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(i-k-1)^2}{2\sigma^2}\right), \quad i \in (-k, k).$$

Vrijednost pojedinog piksela $I'(x_1, y_1)$ u izgladenoj slici računa se, pomoću vektora $h(i)$ duljine $2k + 1$ na sljedeći način:

$$I'(x_1, y_1) = \sum_{i=-k}^k I(x_1 + i, y_1) * h(i) + \sum_{i=-k}^k I(x_1, y_1 + i) * h(i)$$

Rezultat glađenja u pojedinom pikselu je linearna kombinacija vrijednosti „ne-izgladenog“ piksela i njegovih $2k$ susjednih piksela. Slika 2.2 prikazuje zaglađivanje prethodno istaknutih rubova.



Slika 2.2 – Primjer zaglađivanja rubova. Crvenom bojom je označen uvećan dio iz prethodno obrađene verzije bez zaglađivanja (1), a plavom iz obrađene verzije sa zaglađivanjem (2).

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

2.3. Programska implementacija početne obrade slike

Početna obrada slike je implementirana u prototipu koji osim obrade služi i za detekciju Harrisovih kutova (implementacija detekcije je obrađena posebno u sljedećem poglavlju). Prototip je napisan u Pythonu (v2.7), skriptnom jeziku, i poslužio je kao svojevrsno upoznavanje s projektom i njegovom zahtjevnosti. Završna verzija, napisana u C++ jeziku, sadrži funkcionalnost prototipa i algoritmi koji se koriste su skoro identični (njihovi pseudokodovi su dolje navedeni). Primjeri obrade slike koji su prethodno prikazani (Slika 2, 2.1 i 2.2) su rezultat prototipa (bez detekcije značajki).

Pseudokod algoritama obrade slike (izdvojeno je najbitnije):

- Pseudokod isticanja rubova uz pomoć Sobelovog operatora:

Računanje gradijenta za X - os:

```

Stvoriti potrebne kernele Sobelovog operatora;
Izračunati prvu konvoluciju;
Množiti vrijednosti piksela s prvim kernelom;
Stvoriti praznu matricu gradijenata po X - osi;
Izračunati drugu konvoluciju;
Množiti vrijednosti piksela s drugim kernelom;

```

Ponoviti prethodno opisan postupak za Y - os;

- Pseudokod zaglađivanja rubova uz pomoć Gaussovog filtra:

```

Izračunati duljinu i stvoriti vektor te duljine;
Izračunati iznose Gaussove funkcije i središnji član;
Izračunati prvu konvoluciju vektora i originalne slike po X - osi;
Izračunati drugu konvoluciju vektora s prvom konvolucijom po
Y - osi ;

```

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

3. Harrisovi kutovi kao značajke slike

Postoje različite vrste značajki slike, a zajedničko im je što pružaju bitne informacije slike koje služe za određene operacije kojima se dolazi do rješenja. Značajke koje se koriste za praćenje u ovom projektu se nazivaju Harrisovi kutovi. Slika 3 prikazuje nađene značajke na jednom primjeru.



Slika 3 – Nađeni Harrisovi kutovi se prikazuju crvenim rombovima u pregledniku i rezultatu (video datoteci koja uz originalni naslov sadrži '(pracenje)') završnog programa.

3.1. Harrisovi kutovi

Postupak je predstavljen 1988. godine, ali još uvijek predstavlja jedan od najefikasnijih pristupa s fiksnim mjerilom za detekciju značajki. Ideja algoritma je pronaći točke koje imaju maksimalnu različitost susjedstva u ovisnosti o zadanom pomaku. Autokorelacijska formula koja određuje različitost glasi:

$$Eq(d) = \sum_{x,y \in W(q)} [I(x + dx, y + dy) - I(x, y)]^2$$

$Eq(d)$ je vrijednost različitosti koju tražimo.

d je zadani pomak, a dx i dy su njegove komponente.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Tražimo značajke kojima $E(q)$ raste u svim smjerovima, tj. traži se okno s velikim promjenama intenziteta. Izraz $I(x + dx, y + dy)$ može se aproksimirati Taylorovim redom gdje su I_x i I_y parcijalne derivacije. Izraz će se prikazati u vektorskom obliku.

$$\sum_{q \in W} [I(q) + d - I(q)]^2$$

Nakon već spomenutog razvoja po Taylorovom redu dobiva se:

$$\sum_{q \in W} [I(q) + \nabla I(q) \cdot d - I(q)]^2$$

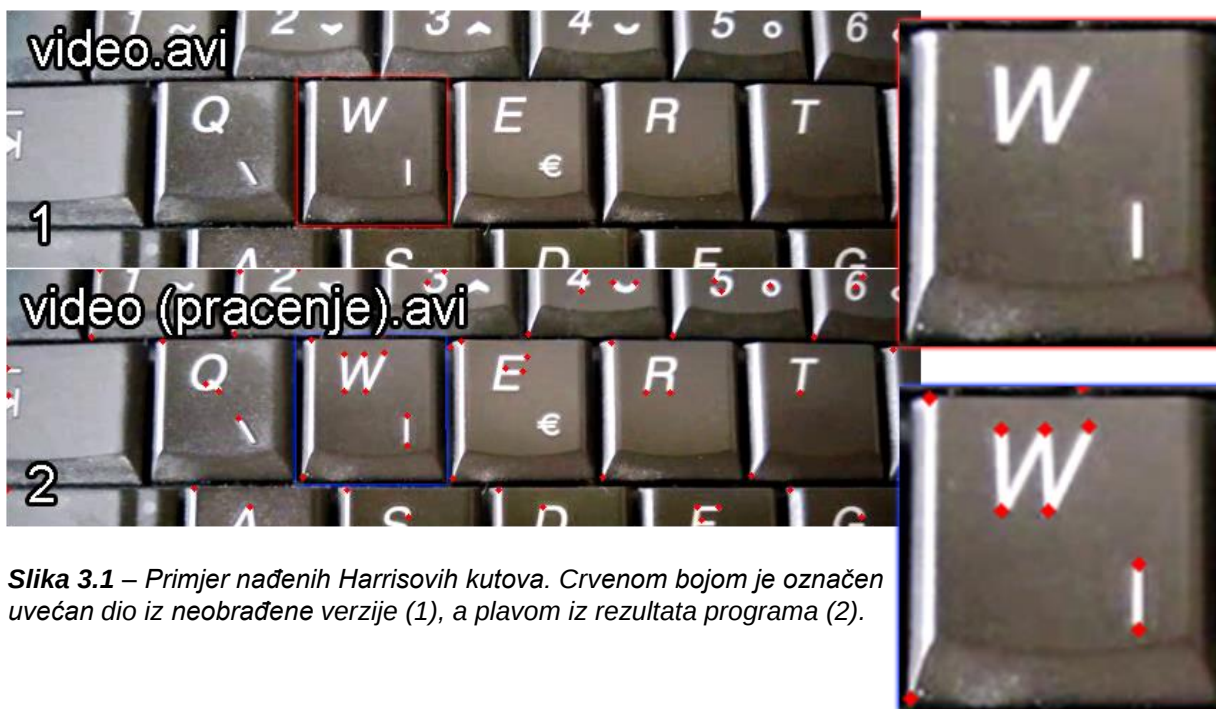
Formulu također možemo prikazati u matričnom obliku:

$$M = \sum_w(x, y) \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$

Tada jednačina izgleda:

$$E(dx, dy) = \begin{pmatrix} dx & dy \end{pmatrix} M \begin{pmatrix} dx \\ dy \end{pmatrix}$$

Za svako detekcijsko okno računa se odziv i određujemo sadrži li kut odziv veći od zadanog praga. Ako je odziv veći od zadanog praga okno se smatra kutom. Slika 3.1 prikazuje nađene Harrisove kutove na jednom primjeru.



Slika 3.1 – Primjer nađenih Harrisovih kutova. Crvenom bojom je označen uvećan dio iz neobrađene verzije (1), a plavom iz rezultata programa (2).

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

3.2. Programska implementacija detektora Harrisovih kutova

Detekcija Harrisovih kutova je implementirana u prototipu, a tu sposobnost sadrži i završni program (uz korištenje sličnog algoritma prikazanog pseudokodom). Pseudokod algoritma detektora Harrisovih kutova:

```
Pomoću Sobelovog operatora izračunati gradijente (X i Y -osi);
Pomoću Gaussovog filtra zagladiti sliku i izračunati konvolucije;
Izračunati trag i determinantu matrice;
Pomoću svojstvenih vrijednosti izračunati odaziv kuta (veća
vrijednost = veća vjerojatnost da je zapravo kut);
Uspoređivanje s okolnim susjedima u okolici prozora:

    Kut se ne sprema ukoliko susjedni kut ima veću
    vrijednost;
```

Slika 3.2 prikazuje izgled korisničkog sučelja prototipa i nađene značajke na primjeru.



Slika 3.2 – Korisničko sučelje prototipa i njegov prikaz nađenih Harrisovih kutova za određenu veličinu prozora.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Značajke se u slučaju prototipa prikazuju crvenim križićima. Interaktivno se određuje veličina prozora nakon čega prototip određuje koje kutove treba prikazati i zatim ih prikazuje (ako postoji spremljen rezultat za određenu sliku i veličinu prozora, prikazuje taj rezultat), a njihov broj se ispisuje pokraj veličine prozora.

Python, koji se koristio u izradi prototipa, je iznimno popularan programski jezik. Zbog jednostavnosti pisanja, intuitivnosti, preglednosti i bogate biblioteke (i velike zajednice koja nastoji što više proširiti tu biblioteku i općenito poboljšati mogućnost koje Python nudi) privlači ljude programiranju i upoznaje ih s osnovnim principima programiranja. Navedeno je samo dio velikih mogućnosti koje ovaj jezik nudi jer koriste ga i mnogo iskusniji programeri u zadacima koje ne zahtijevaju veliku brzinu izvođenja. Python je skriptni jezik visokog nivoa i kao takav je, u usporedbi s popularnim jezicima niskog nivoa (C, Java, C++ itd.), spor u izvođenju operacija. Jednostavan primjer je izvođenje prototipa prikazano na slici 3.2 gdje je bilo potrebno oko 30 sekundi za nalaženje liste svih potencijalnih kutova (iako, uz veće optimiziranje se to vrijeme vjerojatno moglo malo smanjiti). U KLT pratitelju se koristi periodičko detektiranje Harrisovih kutova u nizu slika što znači da bi praćenje značajki zahtijevalo mnogo vremena. To je ujedno i glavni razlog zašto je završni program napisan u jeziku C++, programskom jeziku niske razine, koji omogućava praćenje značajki, ovisno o rezoluciji slike, u stvarnom vremenu ili razumnom vremenskom roku.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

4. Algoritam KLT pratitelja

Praćenje značajki između slika ima brojne primjene u području računalnog vida, a kako je većina tada poznatih algoritama bila spora, 1981. godine Bruce D. Lucas i Takeo Kanade kreiraju novi postupak koji se bazira na diferencijalnom praćenju optičkog toka, za koji pretpostavlja da je konstantan u okolici promatranog piksela te metodom najmanjeg kvadrata rješava jednadžbe optičkog toka, a namjena mu je poravnavanje predloška i ulazne slike što ostvaruje pronalaskom zajedničkih značajki na obje slike. Na temelju tog postupka, kasnije je u suradnji s Carlom Tomasijem razvijen postupak za praćenje značajki nazvan Kanade-Lucas-Tomasi pratitelj značajki (skraćeno KLT pratitelj). Jednostavna verzija algoritma KLT pratitelja će učitati početnu sliku, na njoj pronaći značajke koje se prate te ih spremi, zatim učitati slijedeću sliku, pomoću Lucas-Kanade metode iterativno pronaći na njoj te iste značajke sa prethodne slike, obustaviti traženje izgubljenih značajki i spremi podatke o svim značajkama koje prati. Nakon što je obavljen algoritam za dvije slike, početna slika se zamjenjuje nasljednikom, te se kao druga slika učitava slijedeća te se takav postupak ponavlja do kraja video zapisa ili niza slika nad kojima se obavlja praćenje. U slučaju većeg niza slika, zbog konstantnog gubitka postojećih (npr. napuštanje kadra) te pojavljivanja novih značajki, detektiranje novih značajki će se uobičajeno pozivati nakon određenog broja slika te dodavati nove značajke u postupak praćenja.

Značajke (značajne informacije) se mogu određivati na različite načine, a u implementaciji ovog projekta se koriste Harrisovi kutovi nađeni Harrisovim detektorom. Postupci obrade slike i detekcije značajki se mogu pročitati u prethodnim poglavljima.

Slijedi algoritam KLT pratitelja:

- Označiti prvu sliku simbolom I , drugu sliku simbolom J , a lokaciju značajke pronađenu na prvoj slici vektorom $\bar{u} = [u_x \ u_y]^T$. Ključni dio samog algoritma je pronalazak značajke u , koju smo detektirali na prethodnoj slici i na slijedećoj slici. Taj pronalazak se vrši tako da se nalazi prozor na drugoj slici koji je dovoljno sličan prozoru na prvoj slici koji sadrži traženu značajku. Sličnost na slici se definira pronalaskom vektora pomaka

$$\bar{d} = [d_x \ d_y]^T$$

za koji će funkcija $\varepsilon(d_x, d_y)$ biti minimalna.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

$$\varepsilon(d_x, d_y) = \sum_{x=u_x-\omega_x}^{u_x+\omega_x} \sum_{y=u_y-\omega_y}^{u_y+\omega_y} (I(x, y) - J(x + d_x, y + d_y))$$

ω_x te ω_y predstavljaju polovine veličine prozora, odnosno prozor je velik točno $(2\omega_x + 1) \times (2\omega_y + 1)$ piksela.

- Postupak dobivanja navedenog vektora:
 - Izračunati parcijalne derivacije slike I po x i y osima:

$$I_x(x, y) = \frac{I(x + 1, y) - I(x - 1, y)}{2}$$

$$I_y(x, y) = \frac{I(x, y + 1) - I(x, y - 1)}{2}$$

- Kreirati matricu prostornih gradijenata H:

$$H = \sum_{x=u_x-\omega_x}^{u_x+\omega_x} \sum_{y=u_y-\omega_y}^{u_y+\omega_y} \begin{bmatrix} I_x^2(x, y) & I_x(x, y)I_y(x, y) \\ I_x(x, y)I_y(x, y) & I_y^2(x, y) \end{bmatrix}$$

- Incijalizirati vektor pomaka te postaviti vrijednosti na 0:

$$\bar{d} = [0 \quad 0]^T$$

- Pokrenuti iterativan postupak traženja pomaka, koji se izvodi najviše ograničen broj puta (maksimalan broj iteracija - K) ili dok iznos konvergencije pomaka ne bude manji od graničnog nakon čega staje izvođenje te se zaključuje da je značajka pronađena.
- Izračunati razliku između početne značajke i pretpostavljene lokacije značajke na drugoj slici:

$$\delta I_k(x, y) = I(x, y) - J(x + d_x, y + d_y)$$

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

- Izračunati vektor pogreške:

$$\bar{b}_k = \sum_{x=u_x-\omega_x}^{u_x+\omega_x} \sum_{y=u_y-\omega_y}^{u_y+\omega_y} \begin{bmatrix} \delta I_k(x, y) I_x(x, y) \\ \delta I_k(x, y) I_y(x, y) \end{bmatrix}$$

- Dobiveni vektor pogreške se pomnoži s inverzom prostorne matrice kako bi se dobili iznosi konvergencije pomaka. Iznos konvergencije se dodaje na dosadašnji pomak:

$$\begin{aligned} \bar{\eta}_k &= H^{-1} \bar{b}_k \\ \bar{d}_k &= d_{k-1} + \bar{\eta}_k \end{aligned}$$

- Ako je postupak u nekom trenutku iskonvergirao, može se zaključiti da je pronađena značajka na slici J te je ona određena sljedećim vektorom:

$$v = u + d$$

- Ukoliko postupak nije iskonvergirao, može se zaključiti da je završio zbog prekoračenja maksimalnog broja iteracija. Vektor v može, ali i ne mora pokazivati na značajku te takvu analizu smatramo neuspješnom, a značajku izgubljenom.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

5. Programska implementacija algoritma KLT pratitelja

Sposobnost prototipa (detekcija Harrisovih kutova) je implementirana u završnom programu koji je napisan u jeziku C++, programskom jeziku opće namjene. Osim navedenog, završni program ima sposobnost praćenja značajki, a to postiže, algoritmom KLT pratitelja koji je naveden u prethodnom poglavlju. Postoje dva osnovna načina rada završnog programa:

- Praćenje značajki između dvije slike u kojem se razlikuju također dva načina rada:
 - Praćenje ručno odabrane značajke
 - Praćenje detektiranih značajki (Harrisovih kutova)
- Praćenje značajki u video datoteci

Navedeni načini rada imaju vlastita jednostavna korisnička sučelja i opisani su u sljedećim podpoglavljima. Do određenog načina rada se dolazi izbornicima završnog programa što je detaljnije objašnjeno u osmom poglavlju 'Upute za korištenje'.

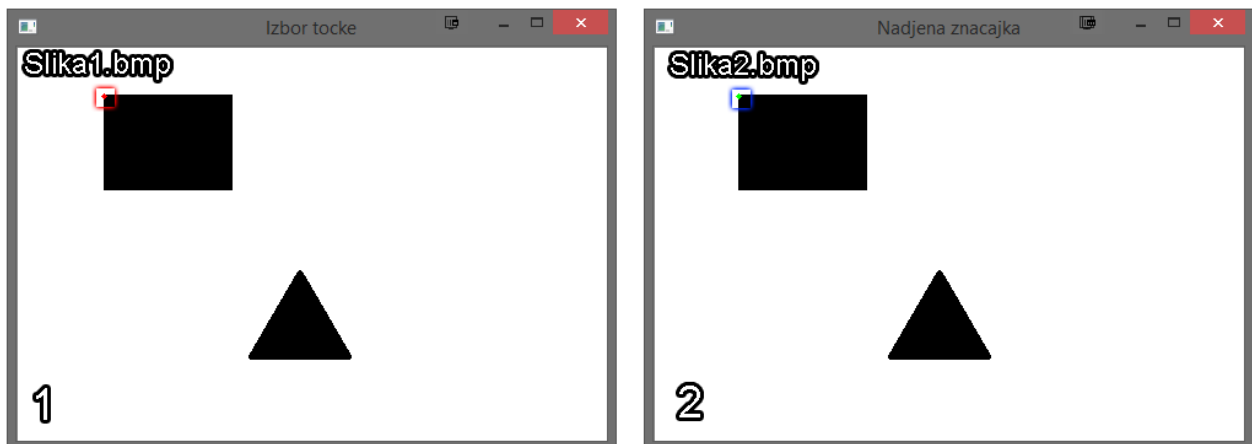
5.1. Praćenje značajki između dvije slike

Prije implementiranja algoritma KLT pratitelja u svrhu praćenja značajki u video datoteci potrebno je bilo testirati navedeni algoritam u slučaju dvije slike u kojima se može lakše pratiti rezultat izvedbe. Kao testni primjer su se koristile dvije jednostavne slike koje imaju istaknuti crni trokut i pravokutnik na bijeloj podlozi. Ti geometrijski likovi su idealni za promatranje praćenja Harrisovih kutova. Razlike u položaju odnosno pomak između slika iznosi nekoliko piksela.

Praćenje značajki između dvije slike se može odvijati na dva načina. Jedan od njih je ručno označavanje značajke koju želimo pratiti. U tom načinu se ne koristi detektor Harrisovih kutova već se prati ručno označena značajka.

Slika 5.1.1 prikazuje korisničko sučelje završnog programa u kojem se prati ručno odabrana značajka.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

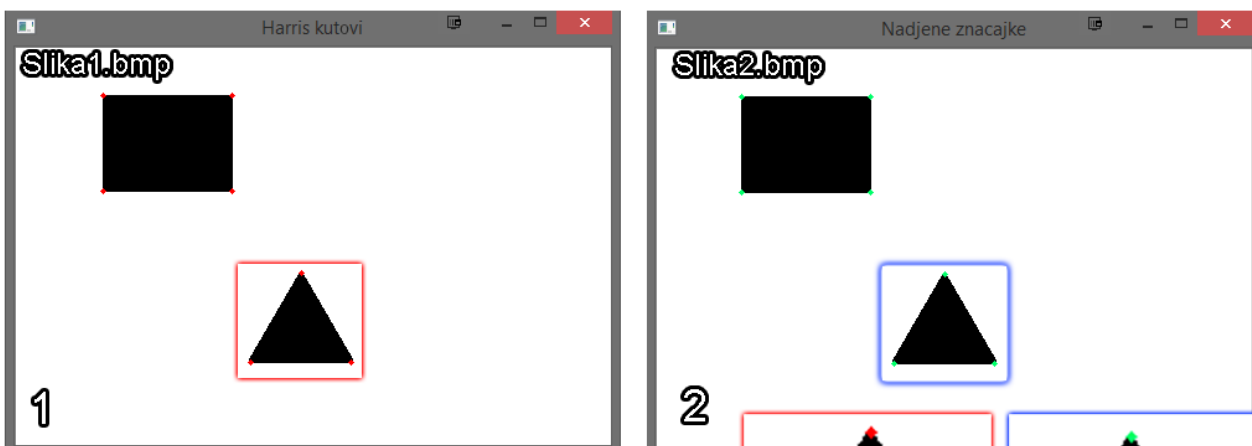


Slika 5.1.1 – U prozoru 'Izbor tocke' (1) se klikom lijevog miša odabere značajka i zatim se u prozoru 'Nadjena znacajka' (2) prikazuje značajka nađena algoritmom KLT pratitelja. Odabrana značajka je označena crvenim, a nađena značajka zelenim romбом (uvećano zbog preglednosti).

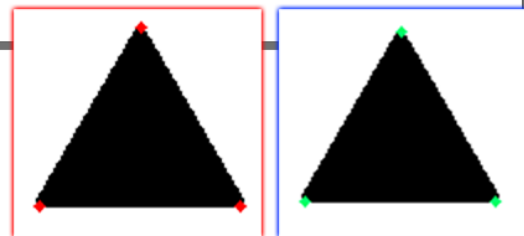


Drugi način je praćenje automatski nađenih značajki. Umjesto praćenja ručno označene značajke, u ovom slučaju se prate nađeni Harrisovi kutovi na prvoj slici.

Slika 5.1.2 prikazuje korisničko sučelje završnog programa u kojem se prati automatski nađene značajke.



Slika 5.1.2 – U prozoru 'Harris kutovi' (1) se prikazuju nađene značajke nađene Harrisovim detektorom i zatim se u prozoru 'Nadjene znacajke' (2) prikazuju značajke nađene algoritmom KLT pratitelja. Praćenje dijela je uvećano i označeno crvenom i plavom bojom zbog preglednosti.



Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

5.2. Praćenje značajki u video datoteci

Video datoteku možemo promatrati kao niz slika (eng. *frames*). Zbog prirode ljudskog vida ljudski mozak može percipirati 10 do 12 individualnih slika u jednoj sekundi no povećanjem tog broja (FPS-a)³ se dobiva osjećaj pokreta. Daljnjim povećanjem FPS-a se povećava fluidnost pokreta što uzrokuje subjektivno ugodniji prikaz. Dogovorno su određeni standardi, koji uzimaju u obzir navedeno uz cijenu izrade i objave video snimki, i iznose 24 (u današnje vrijeme najrašireniji), 25 i 30 slika u sekundi. Zbog razvoja i dostupnosti određenih tehnologija taj će se broj u bližoj budućnosti sigurno povećati.

Princip praćenja značajki uz pomoć KLT algoritma je isti kao za praćenje između dvije slike. Jedina razlika je u tome što se nakon svakih 20 slika koristi detektor Harrisovih kutova tj. u tom slučaju se umjesto praćenja koristi detekcija značajki. Razlog tome je što video snimka može imati naglu promjenu slike (primjer je novi kadar u filmu ili brzi pokret prikazanog lika). Bez periodične detekcije značajke se u tom slučaju mogu izgubiti i njihovo praćenje će biti obustavljeno jer jednom odbačene značajke se više ne uzimaju u obzir.

Korisničko sučelje završnog programa u načinu rada praćenja značajki u video datoteci sadrži tri postavke koje se mogu mijenjati usred praćenja:

- Broj značajki pod postavkom 'Znacajke' u kojoj se može odrediti koliko će se detektiranih kutova odabrati za praćenje. Ovom postavkom se može demonstrirati što bi se dogodilo kad ne bi bilo periodične detekcije značajki (to se može napraviti tako da se usred praćenja broj značajki smanji na nula). Početna vrijednost iznosi 15.
- Prag detekcije Harrisovih kutova pod postavkom 'Harris'. Početna vrijednost iznosi 1000. Detektirane značajke su označene ljubičastim rombovima.
- Prikaz praćenja u kojem program čeka korisnikov upis za prikaz sljedeće slike video datoteke. Početno je ta opcija isključena.

Rezultat završnog programa u ovom načinu je video datoteka istog imena kao ulazna datoteka uz dodatak ' (pracenje)' koja sadrži detektirane i praćene značajke. U slučaju prekida završnog programa usred praćenja, izlazna video datoteka ima trajanje do trenutka prekida.

Slika 5.2 prikazuje korisničko sučelje završnog programa u načinu rada praćenja značajki u video datoteci.

³ FPS je akronim za broj slika u sekundi (eng. *Frames per second*) i koristi se u računalnoj grafici i filmu. Odličan primjer koliko FPS utječe na prikaz slike se može ovdje interaktivno pogledati: <http://goo.gl/mTBDu>

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13



Slika 5.2 – Crvenim rombovima su označene praćene značajke. Na vrhu prozora 'KLT tracking' se mogu vidjeti postavke programa koje se mogu interaktivno mijenjati.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

6. Eksperimentalni rezultati i problemi

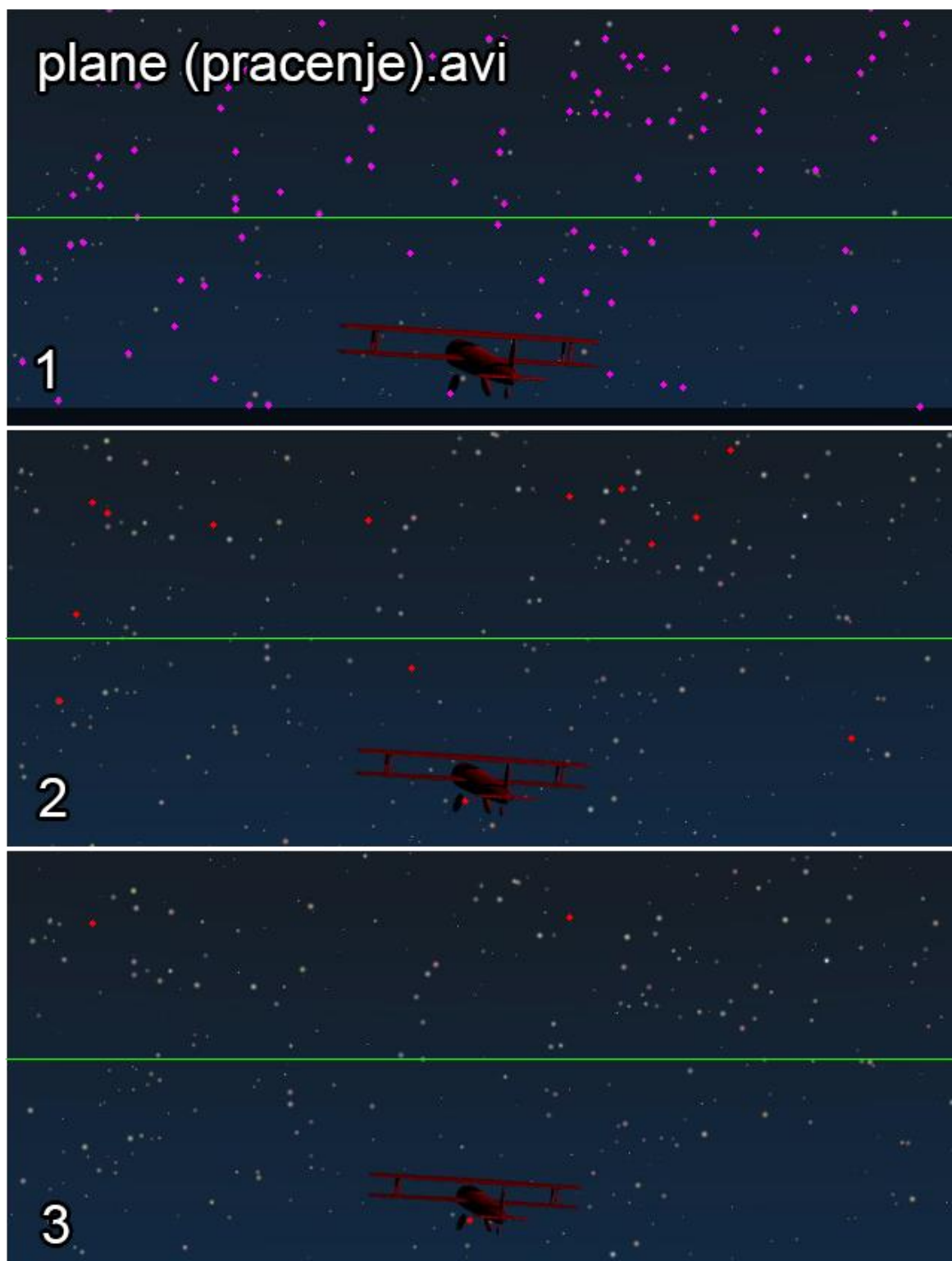
Zadaća ovog projekta je bila pronaći i opisati rješenje problemskog zadatka pod imenom 'Elementarni postupak diferencijalnog praćenja'. Rješenje je nađeno u algoritmu KLT pratitelja. Uz implementaciju navedenog algoritma potrebno je bilo implementirati i početnu obradu slike i detekciju značajki. Posljednje navedeno je prvotno izvedeno u obliku prototipa. Nakon uspješne izrade prototipa izrađuje se završni program koji osim sposobnosti prototipa sadrži i sposobnost praćenja značajki odnosno rješenje u punom obliku. Analizom rezultata završnog programa može se zaključiti da je programska implementacija algoritma KLT pratitelja uspješno napravljena no svakako postoji prostor za doradu i optimizaciju.

Osim poboljšanja izvedbe postoje i određeni problemi. To nisu nužno greške, primjer je vrijeme izvođenja prototipa i završnog programa koje se može vjerojatno malo ubrzati daljnjom optimizacijom no najprije je limitirano programskom i hardverskom platformom. Razlika između izvođenja u programskim jezicima Python i C++ je objašnjena u poglavlju 3.2 Programska implementacija detektora Harrisovih kutova. Primjer hardverskog ograničenja je izvođenje na računalu sa starijim, ali svejedno modernim, dvojezgrenim procesorom Intel Core 2 Duo E8200 (@2.66 GHz) s veličinom spremnika (eng *cache*) od 6 MB. Na njemu se izvođenje praćenja značajki na video, u današnje vrijeme, relativno niske rezolucije od 640x360 piksela (rezolucija primjer video datoteke video.avi) prikazuje u skoro 'stvarnom vremenu' (FPS iznosi oko 25). Za veću rezoluciju od 1280x720 piksela (rezolucija primjer video datoteke spect.avi) to više nije slučaj (FPS iznosi oko 5). Treba napomenuti da je glavni uzrok ovog problema periodična detekcija značajki (Harrisovih kutova) koja zahtijeva mnogo računanja. Rješenje ovog problema je u izradi video datoteke koja se može pokrenuti neovisno o završnom programu i izvodi se kao ulazna video datoteka i sadrži nacrtane značajke. Time je omogućen pregled završnog rezultata u 'stvarnom vremenu' neovisno o hardverskoj platformi.

Standardni algoritam KLT pratitelja (algoritam koji se koristio u ovom projektu) može pratiti značajke koje imaju mali pomak. Za veći pomak potrebno je implementirati tzv. KLT piramidu. Bez upuštanja u detaljniji opis može se ukratko napisati da KLT piramida funkcionira tako što se napravi nekoliko kopija trenutno gledane slike i svakoj kopiji je rezolucija dvostruka manja od prethodne. Izvodi se praćenje između najmanje kopije i slijedeće slike u video datoteci. Pomak značajke se množi dvostruko i zatim se postupak ponavlja za dvostruko veću kopiju, sve dok se ne dođe do originalne slike.

Slika 6 pokazuje primjer rezultata završnog programa u kojem se nađene značajke brzo izgube zbog relativno velikog pomaka.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13



Slika 6 – Prikazani su dijelovi uzastopnih slika iz jednog primjera. Zelenom bojom je označena polovica slika zbog preglednijeg praćenja pomaka. Mogu se vidjeti detektirane značajke označene ljubičastim rombovima (1) koje se ubrzo zbog većeg pomaka gube u sljedećim slikama (2) (3)

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Od još nekih specifičnih slučajeva gdje praćenje ne funkcionira valja spomenuti praćenje značajki koje se pomiču iz slike u sliku po horizontalnoj liniji jednolikih piksela. Algoritmom se određuje što se nalazi u neposrednoj okolini značajke, kako nema razlike između praćene značajke i njezine okoline pomak se ne detektira odnosno značajka se ne prati. Isto vrijedi i za slučaj u kojem se značajka nalazi na površini jednolike boje (piksela).

U rezultatu završnog programa se može primijetiti povremeno 'titranje' nekih značajki u horizontalnom i vertikalnom smjeru. To se događa zbog zaokruživanja koordinata značajki na cijeli broj i dio je algoritma KLT pratitelja.

Detekcija značajki se obavlja svakih 20 slika u video datoteci. Time se osigurava periodično 'obnavljanje' položaja značajki što je korisno u video datotekama u kojima se slike naglo mijenjaju (primjer je promjena kadra u filmu). Uzimajući u obzir da je današnji najčešći standard oko 24 FPS-a u video datotekama, može se zaključiti da se detekcija obavlja periodično manje od jedne sekunde, što je relativno brzo. To nekad nije dovoljno što se može vidjeti u rezultatu za primjer video datoteku 'battle.avi'. To je isječak iz akcijskog filma u kojem se vrlo brzo izmjenjuju kadrovi i može se primijetiti detekcija značajki koja 'kasni' za izmjenom kadra. Daljnje smanjenje tog broja nije preporučljivo jer detekcija značajki uzima veći dio vremena izvedbe.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

7. Zaključak

Ovim projektom se može dobiti iskustvo malog dijela područja računalnog vida. Pritom se ne misli samo na krajnji rezultat već i sve ono što je bilo potrebno napraviti da bi se došlo do toga. Trebalo je, prije svega, razumjeti na koji način računalno zapisuje sliku. Kako se određuje što jest, a što nije nekorisna informacija (šum). Što je to točno značajna informacija slike (značajka) i čemu ona služi. Trebalo je odrediti koja će se vrsta značajki koristiti u ovom projektu. Kojom će se metodom slika obrađivati za što brže nalaženje tih značajki. Zatim je bilo potrebno naći način kojim se smanjuje šum, ali i ne ugrožava korisna informacija slike. Nakon svega trebalo je pronaći način na koji će se pronađene značajke pratiti u nizu slika. Koji su rizici praćenja. Kako unaprijediti krajnji rezultat.

To je sve dio područja računalnog vida. Za potpuno razumijevanje je potrebno dobro predznanje dijela više matematike.

Osnovno diferencijalno praćenje je ovim projektom ostvareno no postoji prostor za nadogradnju kojom bi projekt postao korisniji u nekoj primjeni.

Moguće je dodati određivanje puta neke značajke kojim se ona pomicala. Koliko je bilo trajanje pronađene i praćene značajke. Koliko je značajki potrebno za određivanje pojedinog objekta. Pomoću svega navedenog moguće je odrediti koliko se određeni objekt nalazio u kadru. Jednostavan primjer toga bi bilo praćenje i bilježenje odlaska autobusa na autobusnim stanicama. U istu svrhu bi se mogla pratiti prometovanje određenom ulicom. To je samo jedan od mnogih primjera u kojem bi se nadogradnjom povećala korisnost ovog projekta.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

8. Upute za korištenje programske implementacije

Kako je programska implementacija napravljena u dva dijela posebno će se za svaku napisati što je potrebno za pokretanje i kompiliranje kako bi programi uspješno izvodili ono što je opisano u ovom projektu. U svakom dijelu će biti ukratko napisana struktura izbornika odnosno kako se do njih dolazi. Uz dokumentaciju su priloženi primjeri ulaznih video datoteka: 'battle.avi', 'plane.avi', 'spect.avi', 'video.avi' i 'video2.avi' (zadnja dvije navedene su napravljene u svrhu ovog projekta, a ostale preuzete s interneta). Video datoteke mogu biti ulaz isključivo završnog programa i moraju biti u AVI formatu. Uzrok tome je korištenje biblioteke OpenCV (2.4.6) koja ne podržava druge formate. Također su priložene slikovne datoteke: 'slika1.bmp', 'slika2.bmp' i 'slika3.bmp' (napravljene u svrhu ovog projekta). Slikovne datoteke mogu biti ulazi obje verzije i mogu biti u popularnim formatima (BMP, JPG, PNG). Izlazne datoteke odnosno rezultati ovise o korištenoj verziji programa.

8.1. Prototip: Upute za korištenje

Prototip je napisan u Pythonu, a za njegovo izvođenje i kompiliranje je potrebno Python razvojno okruženje sa sljedećim dodacima:

1. Numpy (za korištenje matrica)
2. OpenCV (za ulazi i izlaz slikovnih datoteka)
3. PIL (za korisničko sučelje)

Python okruženje, 1. i 2. dodatak uz objašnjenje instalacije je moguće ovdje dobiti: <http://goo.gl/6m0nQx>

Modul PIL je moguće dobiti ovdje: <http://goo.gl/DPHxOY> (potrebno je izabrati najnoviju verziju za Python v2.7).

Nakon obavljene instalacije svega potrebnog ulaznu sliku je potrebno staviti u folder 'Input'. Pokretanjem prototipa se pojavljuje izbornik u kojem je potrebno odabrati slikovnu datoteku. Nakon toga se pokreće izvedba prototipa (dvoklikom na ikonu programa ili u Python Shellu) i ispisuju se koraci. Nakon završetka izvedbe pojavljuje se prozor 'Harrisovi kutovi' u kojem je moguće mijenjati veličinu prozora (upíše se broj i pritisne tipka Enter) nakon čega se mijenja prikaz i ispisuje broj kutova. Preporučljivo je izabrati manji broj veličine. Osim što se izlazni rezultat prikazuje u korisničkom sučelju prototipa, on se zapisuje i u folderu Output/harris_temp uz sliku koja sadrži samo značajke u obliku točki.

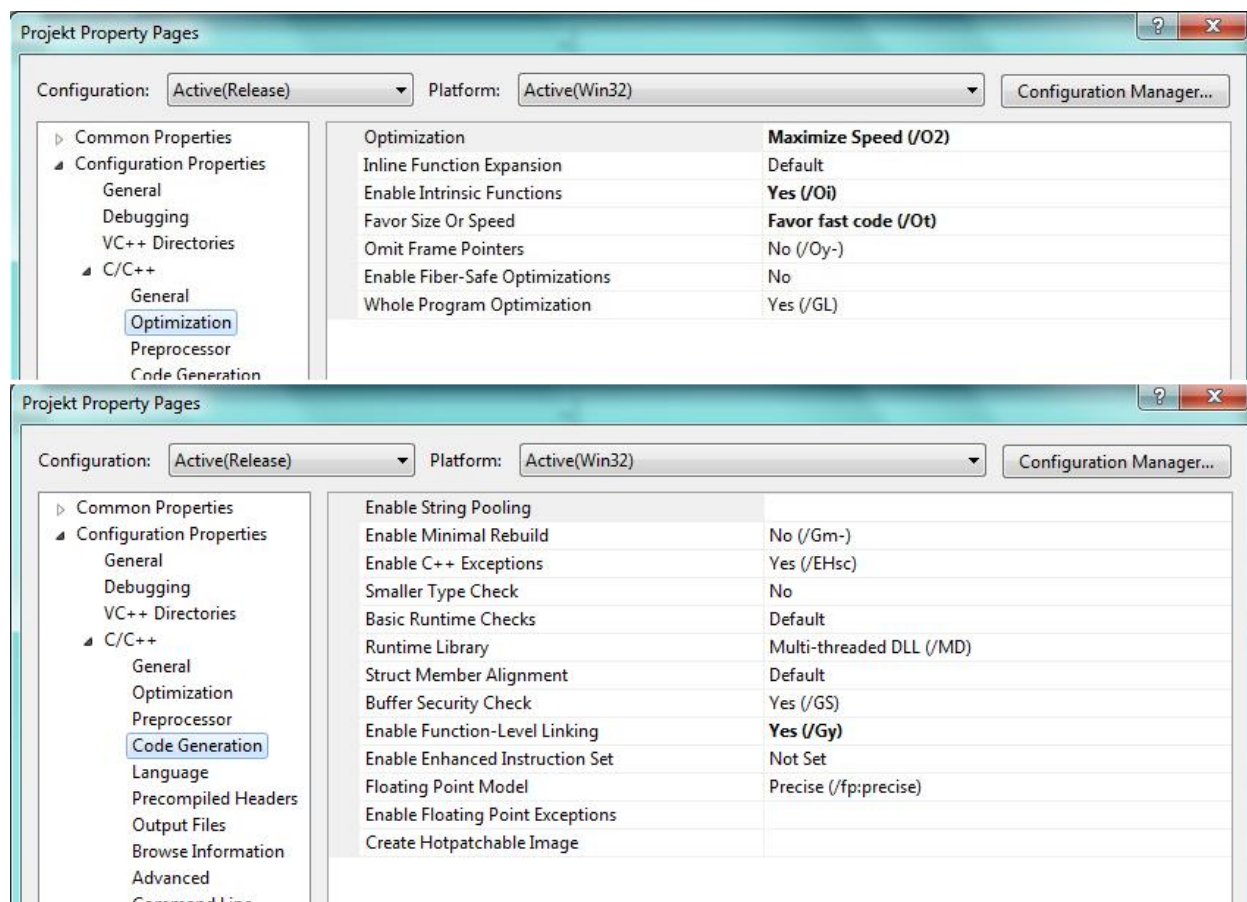
Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

8.2. Završni program: Upute za korištenje

Završni program je napisan u programskom jeziku C++, a za njegovo izvođenje nije potrebna instalacija dodataka, potrebni su jedino datoteke koje su priložene uz kompiliranu verziju. Za kompiliranje završnog programa je potrebno C++ razvojno okruženje (primjer su Visual Studio 20XX za Windows OS i Eclipse za Linux OS) uz korištenje dodatka OpenCV. Za potpuno jamčenje izvedbe projekta savjetuje se koristiti Visual Studio 2010 i OpenCV 2.4.6. jer je programski kod napisan i uspješno testiran u tim verzijama. Objašnjenje instalacije OpenCV biblioteke se može ovdje dobiti:

<http://goo.gl/WQOLkk>

Preporučljivo je uključiti optimizacije kompilatora vezane za povećanje brzine izvođenja izvršnog programskog koda te kompilirati navedeni kod u 'release' verziji. Određivanje optimizacijskih postavki se mogu vidjeti na slici 8.2.



Slika 8.2 – Prikazane su postavke koje treba namjestiti kako bi se ubrzalo izvršavanje programskog koda.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

Pokretanjem završnog programa pojavljuje se glavni izbornik sa sljedećim opcijama:

- Rad sa slikama
- Rad sa videom
- Kraj rada

Pokraj svake opcije je naveden broj s kojim se odabire određena opcija.

U slučaju rada sa slikovnim datotekama pojavljuje se nova lista opcija 'Odaberite format' u kojem se traži format ulazne slikovne datoteke. Odabirom formata se pojavljuje unos za ime prve slike. Potrebno je upisati ime prve slikovne datoteke, pritisnuti tipku Enter i zatim napraviti isto za drugu slikovnu datoteku.

Pojavljaju se dvije opcije:

- Pracenje rucno odabrane znacajke
- Pracenje nadenih Harrisovih kutova

Odabirom opcije se pojavljuju određena korisnička sučelja. U slučaju prve opcije za praćenje značajke je potrebno ručno odabrati značajku na prozoru 'Izbor znacajke' pritiskom na lijevu tipku miša. Zatim se pritiskom tipke Space praćena značajka pojavljuje u prozoru 'Pracena znacajka'. Za povratak u glavni izbornik je potrebno pritisnuti tipku Space.

U slučaju druge opcije se pojavljuje prozor 'Harrisovi kutovi' na kojem su prikazane nađene značajke. Pritiskom tipke Space praćene značajke se pojavljuju u prozoru 'Pracene znacajke'. Za povratak u glavni izbornik je potrebno pritisnuti tipku Space.

U slučaju rada sa video datotekama pojavljuje se unos za ime video datoteke. Potrebno je upisati ime video datoteke i pritisnuti tipku Enter.

Pojavljuje se prozor 'KLT tracking' s postavkama i prostorom za slikovni prikaz.

Izvedba završnog programa se pokreće pritiskom na tipku Space. Postavke se mogu interaktivno mijenjati. Ako se želi samo slikovni prikaz bez postavki potrebno je pritisnuti lijevom tipkom miša na X na vrhu prozora. Za povratak u glavni izbornik je potrebno pritisnuti tipku Escape. Izlaz završnog programa je video datoteka koja u sebi uz ime ulazne datoteke sadrži dio ' (pracenje)' i sadrži detektirane i praćene značajke.

Elementarni postupak diferencijalnog praćenja	Verzija: 1.0
Tehnička dokumentacija	Datum: 06/01/13

9. Literatura

Lucas-Kanade 20 Years On: A Unifying Framework, Baker & Matthews, veljača 2003.

Computer Vision: Algorithms and Applications, Szeliski, rujan 2010.

<http://docs.opencv.org/doc/tutorials/tutorials.html>

http://en.wikipedia.org/wiki/Sobel_operator, prosinac 2013.

http://en.wikipedia.org/wiki/Gaussian_filter, rujan 2013.

http://en.wikipedia.org/wiki/Corner_detection, studeni 2013.