SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 3356

UPRAVLJANJE DIGITALNIM FOTOAPARATOM POMOĆU PROTOKOLA ZA PRIJENOS FOTOGRAFIJA

Zvonimir Mandić

Zagreb, lipanj 2013.

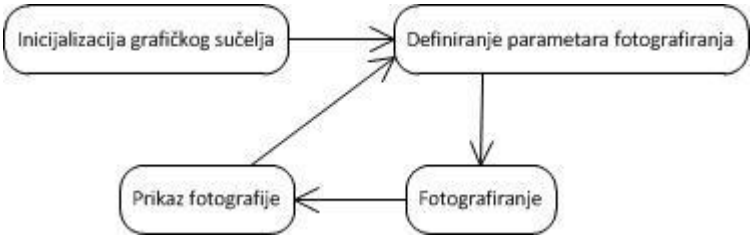
Sadržaj

Uvod

Protokol za prijenos fotografija (engl. Picture transfer protocol, skraćeno PTP) dio je ISO 15470:2008 standarda koji propisuje način komunikacije između fotoaparata i drugih uređaja preko USB-a ili IP mreže bez potrebe za instalacijom dodatnih upravljačkih programa. Trenutna verzija PTP-a je 1.1 i prihvaćena je među svim proizvođačima fotoaparata.

U radu će se koristiti fotoaparat Canon Powershot A570 IS koji s orginalnim *firmwareom* ne podržava operacije PTP-a vezane uz postavljanje parametara za fotografiranje. Na fotoaparat će se postaviti druga verzija *firmwarea* imena CHDK (Canon Hack Development Kit) koja podržava sve operacije PTP-a i dodatno podržava izvođenje Lua skripti na fotoaparatu. Koristeći CHDK, razvit će se aplikacija za upravljanje fotoaparatom sa upravljačkog računala preko USB-a.

Zahtjevi aplikacije za upravljanje fotoaparatom su izvođenje programa na Linux sustavima, grafičko sučelje preko kojeg se unose parametri za fotografiranje i prikazuje fotografija te mogućnost pokretanja drugih programa koji se izvršavaju u komandnoj liniji i komunikacija s njima. Jednostavni dijagram stanja izvođenja programa nalazi se na Slici 1.



Slika 1: Jednostavni dijagram stanja

Rad aplikacije će se testirati na ugradbenom računalnom sustavu Beagleboard s operacijskim sustavom Ångström. Beagleboard je razvila tvrtka Texas Instruments. Bazira se na glavnom procesoru ARM Cortex-A8 i procesoru za obradu digitalnog signala TM320C64x+. Periferiju uređaja čine USB 2.0 priključak, priključci za video S-Video i DVI-D, 3.5 milimetarski stereo priključci za ulazni i izlazni zvučni signal, RS232 priključak i priključak za SD karticu.

Protokol za prijenos fotografija

Protokol za prijenos fotografija (engl. Picture transfer protocol, u daljnjem tekstu PTP) je standardizirani protokol za prijenos fotografija sa digitalnih kamera na druge uređaje. Trenutna verzija PTP-a je 1.1 i standardizirana je u ISO 15470:2008. Namjena PTP-a je prebacivanje slika sa fotoaparata na druge uređaje bez potrebe za instalacijom dodatnih upravljačkih programa. PTP je neovisan o platformi i načinu komunikacije. Fotografije se mogu prebacivati putem USB-a ili TCP/IP mrežom.

PTP je podržan u gotovo svim kamerama od 2005. godine. Na Microsoft operacijskim sustavima PTP je podržan od inačice XP. Apple OS X od verzije 10.1 također podržava PTP. Za Linux operacijske sustave potpora za PTP se ostvaruje instalacijom biblioteka libptp2 ili gphoto.

PTP definira standardne opcije, operacije i odgovore uređaja koje kontroliraju kreiranje objekata poput fotografija. Implementirane opcije imaju dodijeljene jedinstene identifikatore s adresama koje počinju brojem 0x50. Neke od tih opcija su: baterija (ID 0x5001), funkcionalni način rada (ID 0x5002), veličina fotografije (ID 0x5003), postavke kompresije (ID 0x5004), postavke osvijetljenja (ID 0x5005), RGB pojačanje (ID 0x5006), otvor blende (ID 0x5007), žarišna udaljenost (ID 0x5008), udaljenost objekta (ID 0x5009), način fokusiranja (0x500a), postavke ekspozicije (0x500b), bljeskalica (0x500c), vrijeme ekspozicije (0x500d), način rada ekspozicije (0x500e), ISO vrijednost (0x500f), kompenzacija ekspozicije (0x5010) datum i vrijeme (0x5011), kašnjenje okidanja (0x5012), način okidanja (0x5013), kontrast (0x5014), oštrina (0x5015), digitalno povećanje (0x5016), efekti slike (0x5017), broj slika za okidanje u burst načinu rada (0x5018), interval između okidanja u burst načinu rada (0x5019), broj slika za timelapse način rada (0x501a), interval između okidanja u timelapse načinu rada (0x501b), postavke fokusa (0x501c), lokacija za prebacivanje slika (0x501d), ime autora (0x501e), copyright (0x501f).

Vrijednosti opcija osvjetljenja, načina fokusiranja, postavki ekspozicije, bljeskalice, načina rada ekspozicije, načina okidanja, efekata slike i postavki fokusa mogu poprimiti samo unaprijed definirane vrijednosti podržane od proizvođača uređaja.

Vrijednosti opcije RGB pojačanja postavljaju se s tri broja koji predstavljaju omjer crvene, zelene i plave boje u fotografiji.

Vrijednosti opcija otvora blende i žarišne daljine se postavljaju s njihovim vrijednostima pomnoženim sa 100.

Udaljenost objekta se postavlja u milimetrima. Maksimalna udaljenost je 0xFFFF i označava beskonačnu udaljenost.

Vrijeme ekpozicije se postavlja tako da se vrijednost u sekundama pomnoži sa 10000.

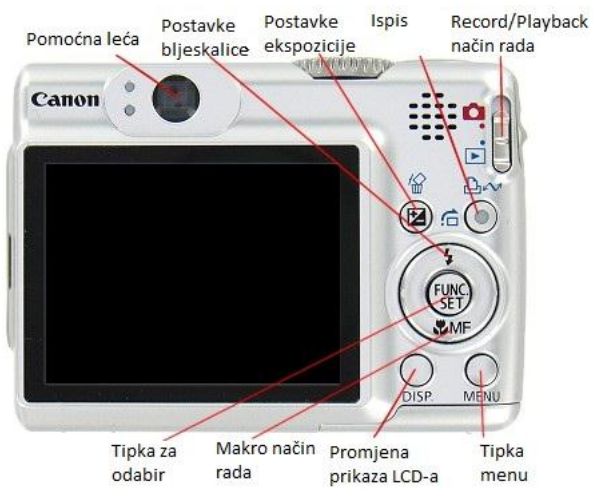
ISO vrijednost ili osjetljivost na svjetlost se postavlja bez mjenjanja. Vrijednost 0xFFFF predstavlja automatsko postavljanje ISO vrijednosti.

Načini postavljanja ostalih opcija nisu točno propisani PTP standardom a zadaju se po njihovoj prirodnoj vrijednosti.

Canon PowerShot A570 IS

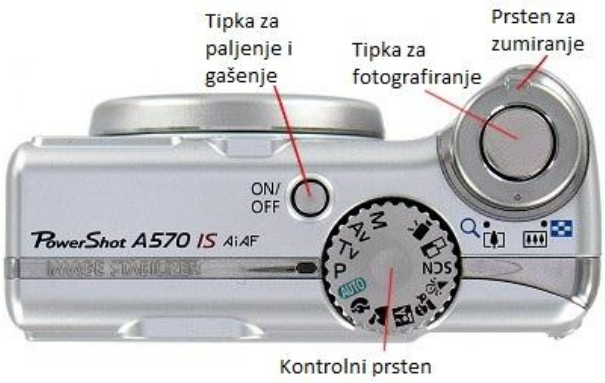
Korišteni fotoaparat je Canon PowerShot A570 IS. A570 je digitalni fotoaparat iz Canonove serije kompaktnih fotoaparata s pristupačnom cijenom. Njegove karakteristike su senzor CCD tipa sa 7.1 milijuna piksela, procesor za obradu slike DIGIC III, optičko povećanje do 4 puta, stabilizator slike i LCD zaslon veličine 2.5 inča.

Slike fotoaparata s označenim dijelovima su na slici 2, 3 i 4.



Slika 2: Stražnja strana fotoaparata

Slika 3: Prednja strana fotoaparata



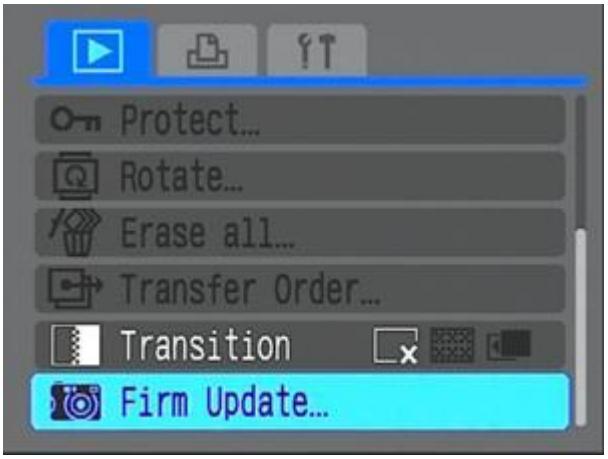
Slika 4: Gornja strana fotoaparata

Canon Hack Development Kit

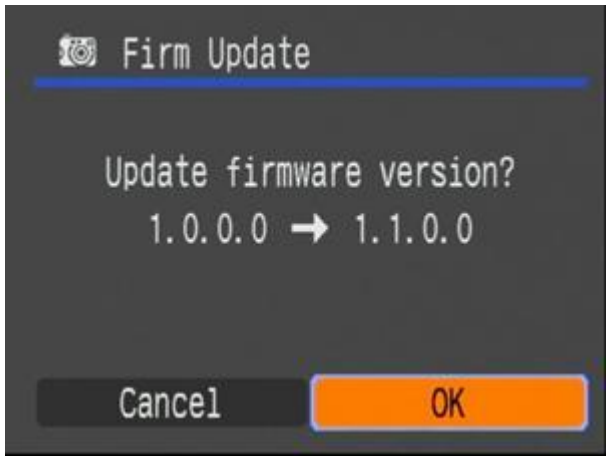
Canon Hack Development Kit (u daljnjem tekstu CHDK) je nadogradnja *firmwarea* za Canonove PowerShot kompaktne fotoaparate. CHDK omogućuje dodatne funkcionalnosti koje nisu podržane orginalnim *firmwareom*. Neke funkcionalnosti su ručno podešavanje parametara okidanja fotografije (brzina okidanja, osjetljivost na svjetlo, otvor blende, udaljenost objekta), spremanje fotografija u RAW formatu, prikaz histograma, opcija zebra načina rada, detekcija pokreta, izvođenje Lua skripti na fotoaparatu. CHDK podržava punu funkcionalnost PTP-a.

Instalacijski paket je dostupan na stranici http://chdk.wikia.com/wiki/Downloads

Nakon instalacije CHDK-a prilikom paljenja potrebno je izvršiti nadogradnju *firmwarea* da bi se CHDK pokrenuo.



Slika 5: Prvi korak nadogradnje *firmwarea*



Slika 6: Drugi korak nadogradnje *firmwarea*

PTP na Linux sustavima

Poznatiji alati za rad s PTP uređajima na Linux sustavima su gPhoto2 i ptpcam. Oba alata podržavaju rad s korištenim fotoaparatom. Ptpcam ima posebno razvijenu podršku za CHDK, stoga će se koristiti u radu.

Instaliranje alata ptpcam

Prije same instalacije ptpcam-a potrebno je instalirati odgovarajuće biblioteke koje koristi ptpcam i koje će biti potrebne za instalaciju ptpcama. Naredbom

```
sudo apt-get install build-essential subversion libusb-dev
lua5.1 liblua5.1 liblua5.1-dev
```

instalirat će se svi potrebni paketi za rad s ptpcamom.

Zatim, potrebno je premjestiti se u mapu gdje se želi smjestiti ptpcam te koristeći subversion skinuti kod za sa subversion repozitorija.

```
svn co
http://subversion.assembla.com/svn/chdkde/trunk/tools/ptpcam/
```

Premjestiti se u datoteku ptpcam naredbom `cd ptpcam` i kompajlirati kod naredbom `make`.

Ptpcam će se instalirati u trenutni direktorij. Pokreće se naredbom `./ptpcam`

Uspješnost instalacije može se provjeriti naredbom `./ptpcam -l`

U ispisu naredbe treba biti naveden fotoaparat spojen na računalo.

CHDK proširenje unutar ptpcama

Ptpcam ima ugrađen CHDK način rada do kojeg se dolazi naredbom `./ptpcam --chdk` pri čemu se otvara interaktivno sučelje za rad s fotoaparatom. Oznaka aktivnosti veze je `<conn>`. Ispis podržanih naredbi može se dobiti naredbom `h`. Od podržanih naredbi najbitnije su:

- `upload <local> <remote>`
- `download <remote> <local>`
- `mode <val>`
- `lua <code>`
- `luar <code>`

Naredba upload prebacuje datoteku s računala na fotoaparat. Parametar local je adresa datoteke na računalu koja se prebacuje. Parametar remote je adresa na koju se datoteka sprema na fotoaparatu.

Naredba download prebacuje datoteku s fotoaparata na računalo. Parametar remote je adresa datoteke na fotoaparatu, a parametar local je adresa na koju će se datoteka spremiti na računalu.

Naredbe upload i download će se koristiti za prebacivanje skripta sa računala na fotoaparat i prebacivanje fotografija s fotoaparata na računalo.

Naredbom mode fotoaparat se prebacuje iz playback načina rada u record način rada i obrnuto. Parametar val može poprimiti vrijednosti 0 ili 1. Nula označava playback način rada, a 1 označava record način rada.

Naredbe lua i luar kao argument primaju lua kod koji će se izvršiti na fotoaparatu. Razlika između te dvije naredbe je slijedeća: naredba lua ne može vratiti povratnu vrijednost, dok naredba luar može.

Aplikacija za upravljanje fotoaparatom

Aplikacija za upravljanje fotoaparatom razvijati će se na operacijskom sustavu Ubuntu verzije 12.04. Rabit će se radno okruženje NetBeans s programskim jezikom Java verzije 7. Za izradu grafičkog sučelja rabit će se alati za dizajniranje sučelja ugrađeni u NetBeans. Rabit će se skriptni programski jezik Perl verzije 5 za komunikaciju između Java programa i programa ptpcam. Za izvođenje skripta na fotoaparatu rabit će se skriptni jezik Lua verzije 5.1. Kao pomoćna skripta za slanje naredbi fotoaparatu i spremanje povratnih vrijednosti dobivenih tijekom izvođenja skripte rabit će se lptpgui.lua skripta dostupna na stranici http://chdk.wikia.com/wiki/PTP_Extension.

Radno okruženje NetBeans preuzeto je sa stranice

https://netbeans.org/downloads/index.html

Podrška za Perl je ugrađena u korišteni operacijski sustav, a podrška za skriptni jezik Lua instalirana je prilikom instalacije ptpcama.

Perl skripte

Perl skripte pokrivaju funkcije vezane uz komunikaciju s fotoaparatom pomoću programa ptpcam i kreiranje izlaznih datoteka. Rabit će se skripte fetchFileList, fetchLocalPictureList, commands i shoot. Izvorni oblik Perl skripti fetchFileList i shoot dostupan je na stranici http://chdk.wikia.com/wiki/PTP_Extension. Skripte su izmjenjene za potrebe ovog rada.

Skripta fetchFileList

Skripta fetchFileList dohvaća popis svih fotografija koji se nalaze u direktoriju DCIM na fotoaparatu. To ostvaruje pozivom funkcije dciml iz skripte lptpgui. Naredba se šalje programu ptpcam preko komandne linije.

```
./ptpcam --chdk="luar require ('lptpgui').dcimdl(false)"
```

Funkcija dciml će kreirati popis fotografija koje se nalaze u direktorij DCIM na fotoaparatu nakon čega perl skripta skida taj popis u radni direktorij u datoteku fileList.txt.

```
./ptpcam --chdk="download A/ptpgui.txt fileList.txt"
```

Zatim iz popisa filtrira imena fotografija i sprema ih u polje.

```
foreach $item (@fileList) {
   ($path,$name,$datetime,$junk)=split(/\|/,$item);
   if(index($path,"101CANON")!=-1) {
      push(@pictureNames,$name);
   }
}
```

Popis sadrži put do datoteke, ime datoteke, datum kreiranja i dodatne poruke. Imena fotografija koja će se spremiti u polje moraju se nalaziti unutar direktorija 101CANON kojeg CHDK koristi za spremanje fotografija.

Iz tog polja će se pronaći zadnja fotografija te će se njeno ime spremiti u datoteku lastPicture.txt. Zadnja fotografija će biti ona fotografija koja ima najveći broj u svom imenu. Ukoliko je ta fotografija obrisana na fotoaparatu, program će dohvatiti krivo ime zadnje fotografije. Nužno je da se zadnja fotografija nalazi spremljena na fotoaparatu. Druge opcije za dohvat zadnje fotografije su bile spremanje brojača fotografija na računalo tako da se pročita broj zadnje fotografije sa ekrana fotoaparata ili dohvat popisa svih fotografija nakon svakog fotografiranja. Za prvu opciju ukoliko bi korisnik fotografirao direktno sa fotoaparata, taj brojač bi se morao ručno mjenjati nakon svakog korištenja. Druga opcija je vremenski zahtjevna stoga se ne koristi.

Skripta fetchFileList se nalazi u dodatku A.

Skripta fetchLocalPictureList

Skripta fetchLocalPictureList traži fotografije po radnom direktoriju i sprema popis njihovih imena u datoteku pictureList.txt.

Popis svih datoteka u radnom direktoriju dohvaća se naredbom `ls` u komandnoj liniji.

```
$returnMessage = `ls`;
```

Iz popisa filtrirat će se datoteke koje počinju slovima IMG i CRW. Pretpostavlja se da neće postojati datoteke ili mape koje počinju istim slovima a nisu fotografije.

Skripta se nalazi u dodatku B.

Skripta commands

Skripta commands prima dva argumenta: broj naredbe i ime datoteke za brisanje. Broj naredbe može biti broj od 1 do 10. Naredbe s pripadajućim brojevima su:

1. Upali LCD-a na fotoaparatu
2. Ugasi LCD-a na fotoaparatu
3. Spremaj slike u RAW formatu
4. Ne spremaj slike u RAW formatu
5. Uključi smanjivanje šuma u RAW fotografijama
6. Isključi smanjivanje šuma u RAW fotografijama
7. Pročitaj temperaturu na fotoaparatu
8. Pročitaj stanje SD kartice
9. Obriši datoteku na računalu
10. Obriši datoteku na fotoparatu

Paljenje LCD-a obavlja se naredbom `./ptpcam --chdk="lua set_backlight(1)"`. Gašenje se obavlja istom naredbom s razlikom u parametru funkcije set_backlight kojeg se postavlja na 0.

Spremanje slike u RAW formatu omogućuje se naredbom `./ptpcam --chdk="lua set_raw(1)"`. Za onemogućavanje spremanja u RAW formatu funkciji set_raw se šalje argument 0.

Smanjivanje šuma se omogućuje naredbom `./ptpcam --chdk="lua set_raw_nr(1)"`. Za isključenje smanjenja šuma funkciji set_raw_nr šalje se parametar 0.

Čitanje temperature fotoaparata obavlja se naredbom `./ptpcam --chdk="luar get_temperature(x)"`. Parametar x može poprimiti vrijednosti 0, 1 ili 2. Za vrijednost 0 funkcija vraća temperaturu na objektivu, za 1 vraća temeperaturu na CCD senzoru a za 2 vraća temperaturu baterije. Povratne vrijednosti temperature spremaju se u trenutni direktorij u datoteku temperature.txt.

Podaci o SD karticu koji se dohvaćaju su: veličine SD kartice, veličina slobodnog prostora na SD kartici te broj JPG i RAW fotografija koje stanu na SD karticu. Veličina SD kartice dohvaća

se naredbom `./ptpcam --chdk="luar get_disk_size()"`. Veličina slobodnog prostora na SD kartici dohvaća se naredbom `./ptpcam --chdk="luar get_free_disk_space()"`. Veličine povratnih vrijednosti tih naredbi izražene su u bajtovima. Broj JPG slika koje stanu na SD karticu dohvaća se naredbom `./ptpcam --chdk="luar get_jpg_count()"` a broj RAW slika dohvaća se naredbom `./ptpcam --chdk="luar get_raw_count()"`. Dohvaćene vrijednosti se spremaju u trenutni direktorij u datoteku SDcard.txt.

Brisanje datoteke na računalu obavlja se naredbom `rm $filename`.

Brisanje datoteke na fotoaparatu obavlja se naredbom `./ptpcam --chdk=\"luar os.remove(\\\"A/DCIM/101CANON/$filename\\\")\"`.

Skripta commands nalazi se u dodatku C.

Skripta shoot

Skripta shoot prima četiri parametra: duljina ekspozicije, broj fotografija, broj zadnje fotografije i zastavica za prebacivanje raw fotografija . Parametri se spremaju u varijable s imenima shutterspeed, numFilesToGet, pictureNumber i raw. Parsiranje parametara iz komandne linije se obavlja sljedećom naredbom:

```
my ($shutterspeed,$numFilesToGet,$pictureNumber,$raw) = @ARGV;
```

Prije početka fotografiranja, skripta će prebaciti Lua skriptu na fotoaparat.

```
./ptpcam --chdk="upload $myScript A/CHDK/SCRIPTS/$myScript"
```

Skripte se prebacuju u direktorij A/CHDK/SCRIPTS na fotoaparatu.

Prije pokretanja skripte na fotoaparatu potrebno je prebaciti se u record način rada.

```
./ptpcam --chdk="mode 1"
```

Zatim se pomoću skripte lptpgui pokreće prebačena Lua skripta.

```
./ptpcam --chdk="lua
require('lptpgui').exec_luafile([[A/CHDK/SCRIPTS/$myScript]])"
```

Nakon pokretanja skripte, potrebno je čekati završetak izvođenja.

```
$scriptStatus = `./ptpcam --chdk="script-status"`;
     ($junk,$run,$msg)=split(/\s+/,$scriptStatus);
     printf("  > status: $run - $msg\n");
```

```
    $wait--; if ($run =~ /no/) {$wait=0;}
```

Status izvođenja skripte može se dobiti naredbom `script-status`. Nakon parsiranja odgovora naredbe provjerava se run dio koji može poprimiti vrijednosti `/yes/` i `/no/`. Gornji isječak koda ponavlja se u petlji dok skripta ne završi s izvođenjem.

Nakon završetka izvođenja skripte fotoaparat obavlja fotografiranje. Fotografiranje se odvija u dvije faze: faza s otvorenom blendom i faza sa zatvorenom blendom. Svaka faza traje koliko je zadano vrijeme ekpozicije. Stoga se nakon detektiranja završetka izvođenja skripte na fotoaparatu izvršava funkcija sleep s trajanjem čekanja zadanim preko argumenta shutterspeed.

Nakon završetka fotografiranja fotografije se prebacuju na računalo. Prvo je potrebno prebaciti se u playback način rada.

```
./ptpcam --chdk="mode 0"
```

Broj fotografija koje će se prebaciti na računalo zapisan je u argumentu numFilesToGet. Ime fotografije koje će se prebaciti kreira se pomoću argumenta pictureNumber. Ako je postavljena zastavica raw prebacit će se i raw fotografija.

```
for ($i=$pictureNumber;$i<=($pictureNumber+$numFilesToGet-
1);$i++) {
     printf("fetching IMG_$i \n");
     $result=`./ptpcam --chdk="download
A/DCIM/101CANON/IMG_$i.JPG IMG_$i.JPG"`;
     sleep 2;
     if ($raw==1) {
       $result=`./ptpcam --chdk="download
A/DCIM/101CANON/CRW_$i.DNG CRW_$i.DNG"`;
       sleep 2;
     }
   }
```

Fotografije koje će se prebaciti na računalo spremaju se u trenutni direktorij s istim imenom kao i na fotoaparatu.

Skripta shoot nalazi se u dodatku D.

Lua skripta

Lua skripta pokriva funkcionalnosti vezane uz fotografiranje i fotoaparat. Te funkcionalnosti su postavljanje vremena ekspozicije, otvora blende, ISO vrijednosti, udaljenosti objekta, opcije paljenja ili gašenja bljeskalice, razina optičkog povećanja, broj fotografija i intervala između dva fotografiranja. Imena funkcija koje se koriste navedena su u tablici 1.

Tablica 1: Korištene Lua funkcije

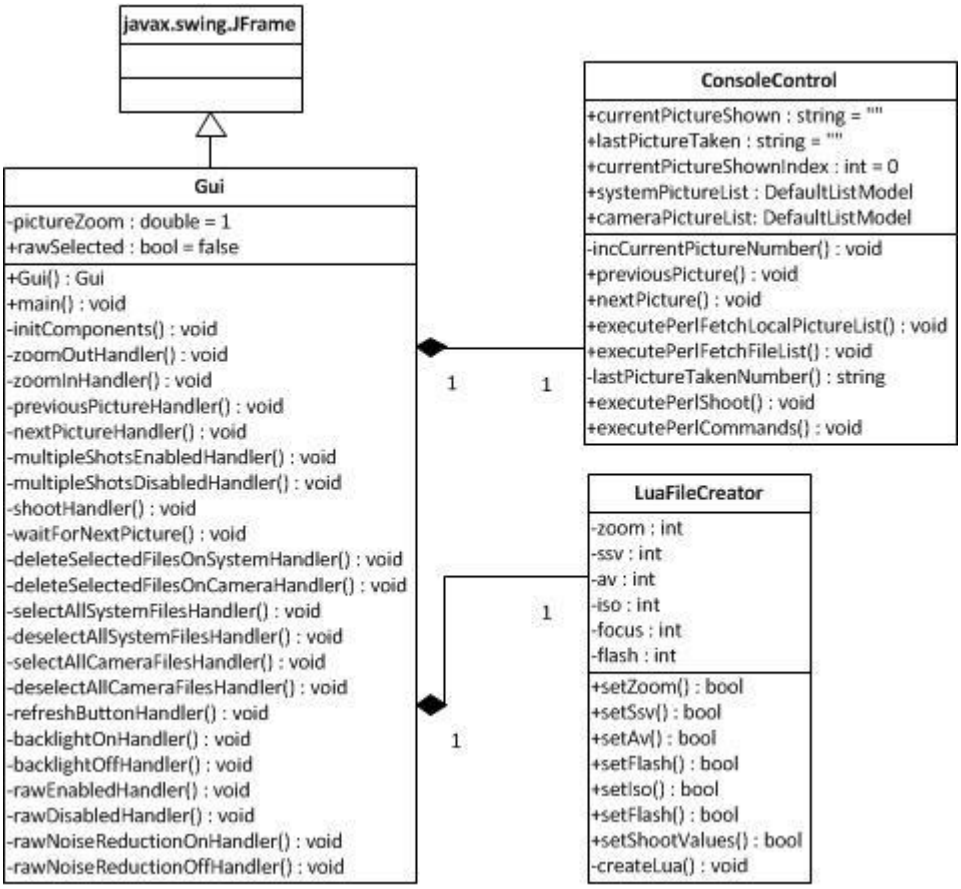
Ime funkcije	**Opis**
zoom(x)	Postavlja razinu optičkog povećanja
set_tv96_direct(x)	Postavlja duljinu ekspozicije
set_av96(x)	Postavlja veličinu otvora blende
set_sv96(x)	Postavlja ISO vrijednost
set_focus(x)	Postavlja udaljenost objekta
set_prop(143,x)	Upravlja bljeskalicom
press “shoot-half”	Simulira pritisak tipke za fotografiranje do pola
press “shoot-full”	Simulira pritisak tipke za fotografiranje do kraja
release “shoot-full”	Simulira otpuštanje tipke za fotografiranje od kraja do pola
release “shoot-half”	Simulira otpupštanje tipke za fotografiranje od pola

Osim funkcija iz tablice 1, u skripti će se koristiti funkcija sleep(x) koja će se biti iza svake funkcije gdje dolazi do promjene parametara fotografiranja. Skripta će također imati varijable loop, interval i counter. Varijabla loop označava koliko puta treba fotografirati. Interval označava koliko treba čekati između fotografiranja. Counter služi kao brojač sekundi (koliko je još potrebno čekati).

Skripta će se kreirati unutar Java koda i spremiti u radni direktorij. Primjer kreirane Lua skripte nalazi se u dodatku E.

Java aplikacija

Java kod pokriva funkcionalnosti kreiranja grafičkog sučelja, rada s grafičkim sučeljem, kreiranja Lua skripte i komunikacije s Perl skriptama. Aplikaciju čine tri razreda: Gui, ConsoleControl i LuaFileCreator. Dijagram razreda nalazi se na slici 7.



Slika 7: Dijagram razreda

Razred Gui

Razred Gui nasljeđuje razred javax.swing.JFrame. U razredu Gui nalaze se metode main, waitForNextPicture, initComponents i metode sa sufiksom Handler koje obrađuju događaje iz grafičkog sučelja. Varijable razreda Gui su: statička varijabla s pomičnim zarezom pictureZoom s inicijalnom vrijednosti 1, statička boolean varijabla rawSelected s inicijalnom vrijednosti false i generirana imena dijelova grafičkog sučelja.

U metodi main inicijalizira se "Look and feel" sučelje i pozivaju se metode executePerlFetchFileList i executePerlFetchLocalPictureList iz razreda ConsoleControl. Metoda executePerlFetchLocalPictureList će postaviti varijablu currentPictureShown iz razreda ConsoleControl koja se koristi u inicijalizaciji grafičkog sučelja. Također, u metodi main poziva se konstruktor razreda Gui koji poziva metodu initComponents.

Metoda initComponents inicijalizira dijelove grafičkog sučelja. U metodi se koristi prethodno postavljena varijabla currentPictureShown iz razreda ConsoleControl da bi se prikazala zadnja fotografija iz lokalnog direktorija. Također se koriste liste systemPictureList i cameraPictureList iz razreda ConsoleControl za ispis fotografija koje se nalaze na fotoaparatu i na računalu.

Veličina fotografije skalira se množenjem širine i visine fotografije s varijablom pictureZoom. Inicijalna vrijednost varijable pictureZoom je jedan. Kod kojim se to ostvaruje je:

```
try {
            BufferedImage img = javax.imageio.ImageIO.read(new
File("./" + ConsoleControl.currentPictureShown));
            int scaleX = (int) (img.getWidth() * pictureZoom);
            int scaleY = (int) (img.getHeight() * pictureZoom);
            Image newImg = img.getScaledInstance(scaleX, scaleY,
Image.SCALE_SMOOTH);
            jLabel2.setIcon(new javax.swing.ImageIcon(newImg));
        } catch (IOException e) {
            System.out.println("Error while loading picture " +
ConsoleControl.currentPictureShown + "\n or no pictures in
workspace");

        }
```

Kod se u istom obliku koristi za postavljanje prikaza fotografije u metodama zoomOutHandler, zoomInHandler, previousPictureHandler, nextPictureHandler, shootHandler.

Grafičko sučelje napravljeno je u alatu za dizajniranje sučelja ugrađenom u radno okruženje NetBeans. Dodavanje novih elemenata obavlja se na „drag and drop“ način. Kod kojim se elementi stvaraju automatski se generira. Parametri elemenata zadaju se unutar sučelja za dizajn. Akcije koje se obavljaju za događaje na tim elementima dodaju se u sučelju za dizajn a uređuju u uređivaču teksta. Akcije će se implementirati na svim tipkama u grafičkom sučelju za događaj actionPerformed. Sučelje je podjeljeno na dva dijela: dio za prikaz fotografije i upravljanje fotografiranjem i dodatni dio gdje se upravlja datotekama, dodatnim opcijama fotoaparata i gdje se ispisuje stanje fotoaparata.

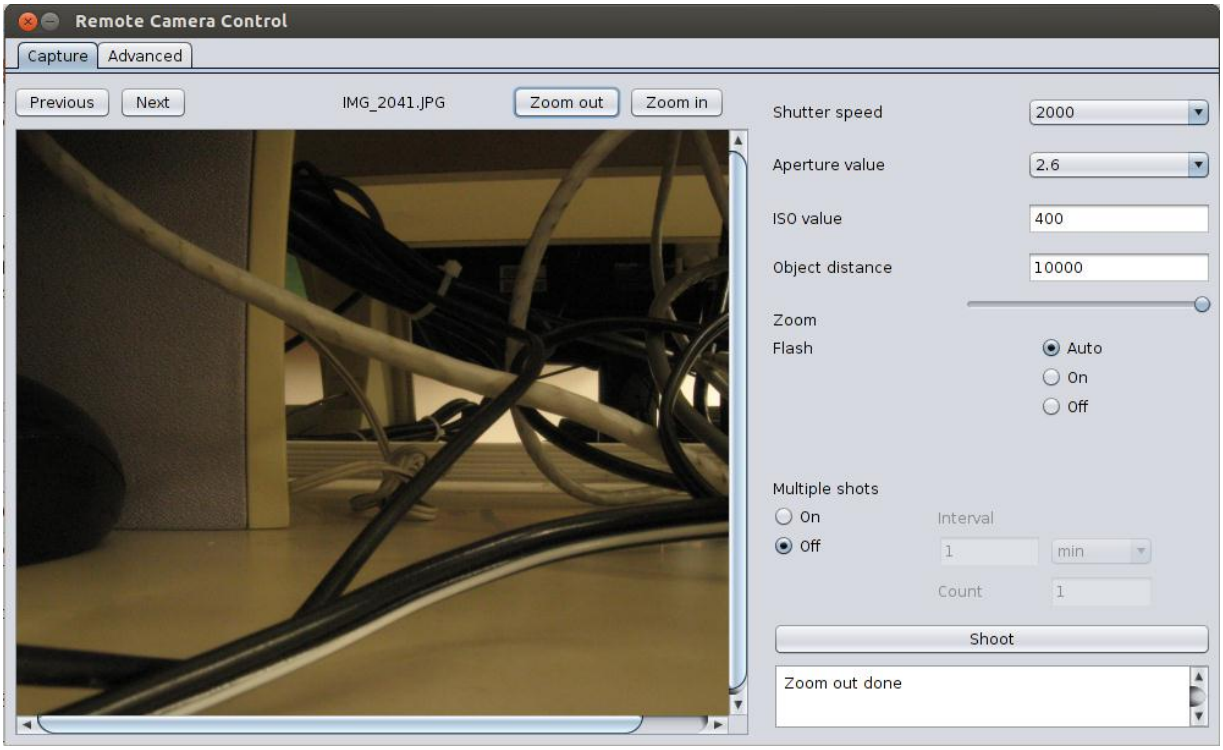
Podjela na dva dijela ostvarena je pomoću komponente jTabbedPane. Komponenta se sastoji od dvije odvojene površine smještene unutar komponenti jLabel kojima se pristupa odabirom oznake Capture ili Advanced.

Dio za prikaz fotografije sastoji se od dijela gdje se fotografija prikazuje i dijela gdje se upravlja prikazivanjem. Fotografija koja će se prikazati nalazi se u komponenti jLabel. Komponenta jLabel smještena je u komponentu jScrollPane u kojoj je automatski omogućeno pomicanje po fotografiji. Za upravljanje prikazom fotografije omogućeno je prikazivanje prošle ili sljedeće fotografije te povećanje ili smanjenje fotografije. Te akcije obavljat će se pritiskom na tipke Previous, Next, Zoom out, Zoom in. Dodatno, u ovom dijelu sučelja ispisivati će se ime prikazane fotografije.

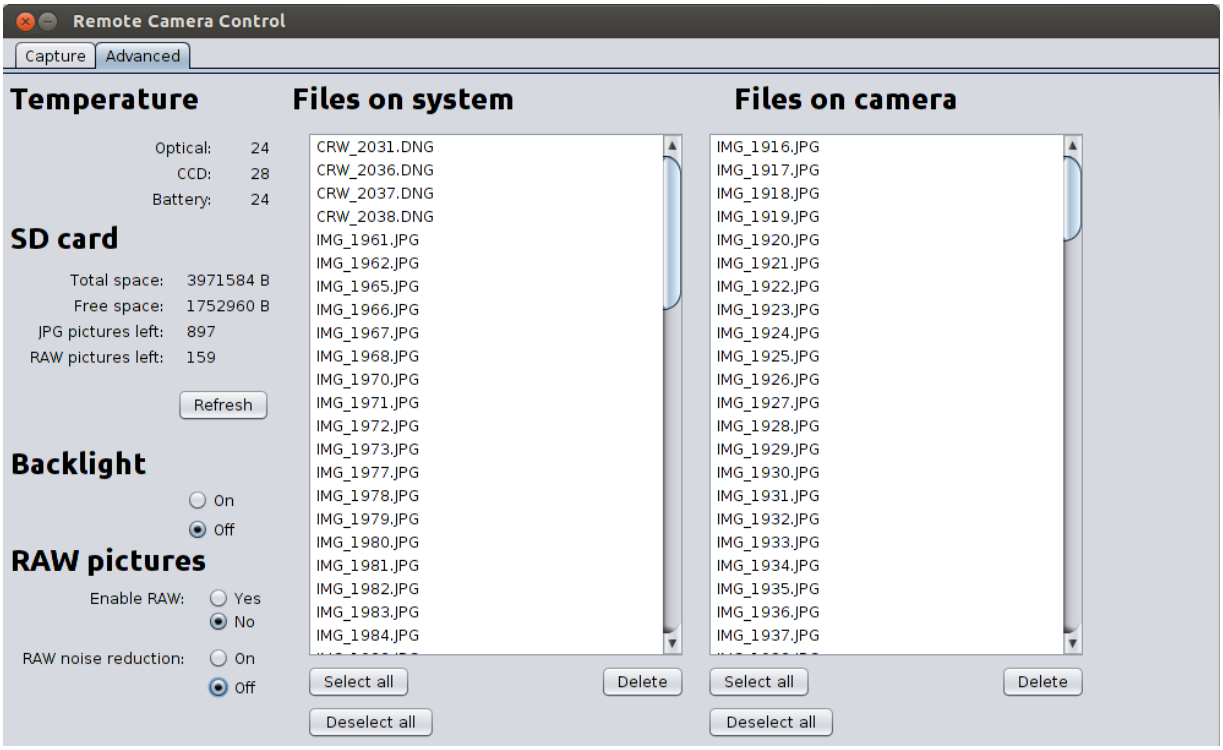
U dijelu za upravljanje fotografiranjem nalaze se padajući izbornici, polja za upis teksta, klizači i radio tipke s labelama koje ih ukratko opisuju. Vrijeme ekpozicije i otvor blende, označeni labelama Shutter speed i Aperture value, zadat će se pomoću padajućih izbornika. Osjetljivost na svjetlinu (ISO value) i udaljenost objekta (Object distance) zadat će se poljima za unos teksta. Razina optičkog povećanja (Zoom) zadat će se preko klizača. Optičko povećanje se na fotoaparatu može zadati u 14 razina stoga se broj razina na klizaču također postavlja na 14. Bljeskalica (Flash) se može zadati da se automatski pali, da se pali prilikom svakog fotografiranja ili da se ne pali. Te opcije će se moći zadati preko radio tipki. Biti će

moguće označiti samo jednu vrijednost. Opcija višetrukog fotografiranja zadat će se preko radio tipki označenih labelom Multiple shots. Interval će se zadati u polju za unos teksta i padajućem izborniku. Broj fotografija zadat će se preko polja za unos teksta označenog labelom Count. Pokretanje fotografiranja obavit će se pritiskom na tipku Shoot. Ispod tipke Shoot nalazit će se polje za ispis teksta u koje će program ispisivati poruke vezane uz izvođenje.

Izgled grafičkog sučelja nakon inicijalizacije nalazi se na slikama 8 i 9.



Slika 8: Izgled grafičkog sučelja



Slika 9: Izgled grafičkog sučelja

Metode za obradu događaja iz grafičkog sučelja implementirane su za događaje na tipkama Shoot, Previous, Next, Zoom in, Zoom out, On, Off, Refresh, na dvije Delete tipke, Select all tipke i Deselect all tipke te na radio tipkama On i Off koje se nalaze uz labele Backlight, Enable RAW i RAW noise reduction. Te metode su implementacije apstraktne metode actionPerformed.

Metode zoomInHandler i zoomOutHandler povećavaju, odnosno, smanjuju prikazanu fotografiju. Prije promjene prikaza, varijabla pictureZoom množi se s faktorom povećanja ili smanjenja veličine fotografije. Faktor povećanja je 1.25 a faktor smanjenja 0.8. Za promjenu prikaza slike koristi se kôd naveden kod metode initComponents.

Metode previousPictureHandler i nextPictureHandler mjenjaju trenutno prikazanu fotografiju na prošlu ili slijedeću. Na početku tih metoda pozivaju se metode previousPicture i nextPicture iz razreda ConsoleControl koje će promjeniti varijablu currentPictureShown. Za promjenu prikaza fotografije koriste kôd naveden kod metode initComponents.

Metode multipleShotsEnabled i multipleShotsDisabled omogućavaju, odnosno, onemogućavaju mjenjanje dijela grafičkog sučelja gdje se postavljaju opcije višestrukog fotografiranja. Kod koji će omogućiti mjenjanje nalazi se u nastavku. Kod koji onemogućuje mjenjanje umjesto argumenata true, šalje argumente false.

```
if (jRadioButton1.isSelected()) {
            jLabel3.setEnabled(true);
            jTextField1.setEnabled(true);
            jComboBox1.setEnabled(true);
            jTextField2.setEnabled(true);
            jLabel5.setEnabled(true);
        }
```

Metoda shootHandler prikuplja podatke unešene u grafičkom sučelju, poziva metodu za kreiranje Lua skripte i metodu za poziv Perl skripte za fotografiranje te prikazuje novu fotografiju. Primjer dohvata vrijednosti iz padajućeg izbornika nalazi se u nastavku.

```
shutter = shutterValues[jComboBox3.getSelectedIndex()];
```

Metoda sadrži lokalne varijable shutterValues i apertureValues koje sadrže sve veličine parametara shutter speed i aperture value koje se mogu zadati u grafičkom sučelju. Također metoda sadrži lokalnu varijablu flag koja ima inicijalnu vrijednost nula. Flag je kontrolna zastavica koja se postavlja ako je odabrana opcija On iz skupa tipki Multiple shots i ako zadani interval iznosi jednu minutu ili manje.

Ako je kontrolna zastavica postavljena, poziva se metoda setShootValues iz razreda LuaFileCreator s argumentima otvora blende, stanja bljeskalice, ISO vrijednosti, duljine ekspozicije, udaljenosti objekta, broju fotografija i intervalom zadanim preko grafičkog sučelja.

```
lua.setShootValues(lua, aperture, flash, iso, shutter,
distance, zoom, numberOfPictures, interval)
```

Zatim se poziva metoda executePerlShoot iz razreda ConsoleControl koja upravlja fotografiranjem. Metodi se šalju argumenti duljine ekspozicije, broja slika te vrijednost kontrolne zastavice.

```
ConsoleControl.executePerlShoot(shutter,
numberOfPictures,flag);
```

Nakon završetka izvođenja metode executePerlShoot, u grafičkom sučelju prikazuje se nova fotografija koristeći kôd naveden kod metode initComponents.

Ako kontrolna zastavica nije postavljena, također se poziva metoda setShootValues s istim parametrima fotografiranja osim parametara broja fotografija i intervala koji se postavljaju na jedan.

```
lua.setShootValues(lua, aperture, flash, iso, shutter,
distance, zoom, 1, 1)
```

Zatim se u metodi ponavlja odsječak koda gdje se poziva metoda executePerlShoot onoliko puta koliko je zadano u polju za unos teksta Count. Ako broj nije zadan, petlja se izvršava samo jednom. Tijekom izvršavanja petlje, nakon poziva metode executePerlShoot, postavlja se nova fotografija i poziva se metoda waitForNextPicture.

```
while(numberOfPictures>0) {
      numberOfPictures--;
      console.executePerlShoot(shutter, 1, flag);
      //prikaz fotografije
      if(numberOfPictures>0) waitForNextPicture(interval);
}
```

Metoda waitForNextPicture prima jedan parametar koji prestavlja duljinu čekanja u minutama. Čekanje se ostvaruje metodom sleep iz razreda Thread.

Metoda deleteSelectedFilesOnSystemHandler implementirana za tipku Delete za datoteke na računalu poziva metodu executePerlCommands iz razreda ConsoleControl s parametrima 9 i imenima datoteke koje će se obrisati. Datoteke označene za brisanje dohvaćaju se naredbom `jList1.getSelectedIndices()`. Metoda getSelectedIndices vraća listu indeksa označenih elemenata liste jList1. Ime datoteke koja će se obrisati dohvaća se naredbom `(String)ConsoleControl.systemPictureList.getElementAt( selectedFilesIndices[i])`.

Nakon brisanja datoteke na računalu potrebno ju je obrisati iz liste. To se ostvaruje naredbom

```
ConsoleControl.systemPictureList.removeElementAt(selectedFilesI
ndices[i]).
```

Metoda deleteSelectedFilesOnCameraHandler je ista kao i gornja metoda, osim što za dohvat označenih datoteka koristi komponentu jList2 i metodi executePerlCommands za prvi parametar šalje broj 10.

Metoda selectAllSystemFilesHandler će označiti za brisanje sve datoteke koje se nalaze na računalu. To se ostvaruje naredbom

```
jList1.addSelectionInterval(0,
ConsoleControl.systemPictureList.getSize()-1)
```

Metoda deselectAllSystemFilesHandler će naredbom `jList1.clearSelection()` maknuti oznake sa svih datoteka koje se nalaze na računalu.

Metode selectAllCameraFilesHandler i deselectAllCameraFilesHandler za razliku od gornje dvije metode pristupaju komponenti jList2 koja se odnosi na datoteke koje se nalaze na fotoaparatu.

Metoda refreshButtonHandler poziva metodu executePerlCommands iz razreda ConsoleControl s argumentima 7 i 8. Nakon završetka izvođenja metode executePerlCommands iz datoteka temperature.txt i SDcard.txt pročitat će se temperature i stanje SD kartice te će se ti podaci ispisati u grafičkom sučelju.

Metoda backlightOnHandler poziva metodu executePerlCommands s argumentom 1 a metoda backlightOffHandler šalje argument 2.

Metode rawEnabledHandler i rawDisabledHandler postavljaju varijablu rawEnabled na true ili false te pozivaju metodu executePerlCommands s argumentom 4 ili 5.

Metode rawNoiseReductionOnHandler i rawNoiseReductionOffHandler pozivaju metodu executePerlCommands s argumentima 5 i 6.

Razred ConsoleControl

Razred ConsoleControl sadrži metode executePerlFetchFileList, executePerlCommands, executePerlFetchLocalPictureList, executePerlShoot, nextPicture, previousPicture, incCurrentPictureNumber, lastPictureTakenNumber i varijable currentPictureShown i lastPictureTaken.

Metoda incCurrentPictureNumber kao argument prima iznos za koliko će povećati broj uz ime fotografije zapisane u varijablama currentPictureShown i lastPictureTaken. Kod koji će povećati broj fotografije za jedan nalazi se u nastavku.

```
String delims = "[_.]+";
String[] tokens = currentPictureShown.split(delims);
String[] tokens2 = lastPictureTaken.split(delims);

currentPictureShown =
"IMG_"+String.valueOf(Integer.valueOf(tokens[1])+1)+".JPG";
lastPictureTaken =
"IMG_"+String.valueOf(Integer.valueOf(tokens2[1])+1)+".JPG";
```

Novo ime fotografije dodaje se u liste systemPictureList i cameraPictureList naredbama

```
ConsoleControl.systemPictureList.addElement(
currentPictureShown)
ConsoleControl.cameraPictureList.addElement(
currentPictureShown)
```

Ako je označeno spremanje fotografija u RAW formatu u liste se dodaje i RAW fotografija i to naredbama:

```
ConsoleControl.systemPictureList.addElement(
"CRW_"+pictureNumber+".DNG")
ConsoleControl.cameraPictureList.addElement(
"CRW_"+pictureNumber+".DNG")
```

Kod se ponavlja u petlji onoliko puta koliko je zadano argumentom.

Metode nextPicture i previousPicture mjenjaju vrijednost varijable currentPictureShown na slijedeće ili prošlo ime prema popisu imena fotografija koji se nalazi u listi systemPictureList. Isječak koda iz metode previousPicture koji će pronaći prošlu fotografiju nalazi se u nastavku.

```
if(currentPictureShownIndex==0) {
currentPictureShownIndex = systemPictureList.getSize()-1;
while((currentPictureShown=
      (String)systemPictureList.getElementAt(
currentPictureShownIndex)).contains("CRW")) {
     currentPictureShownIndex--;
     return;
     }
```

```
}
currentPictureShownIndex--;
while((currentPictureShown=(String)
systemPictureList.getElementAt(
currentPictureShownIndex)).contains("CRW")) {
               currentPictureShownIndex--;
                    return;
}
```

Metoda executePerlFetchLocalPictureList poziva Perl skriptu fetchLocalPictureList i postavlja varijable currentPictureShown, currentPictureShownIndex i systemPictureList. Za postavljanje varijabli koristi se datoteka pictureList.txt koju kreira Perl skripta. Vrijednost na koju se postavlja varijabla currentPictureShown jednaka je zadnjem zapisu u toj datoteci. Svi elementi datoteke se dodaju u listu systemPictureList. Isječak koda koji ostvaruje navedene funkcionalnosti nalazi se u nastavku.

```
BufferedReader reader = new BufferedReader(new
FileReader("pictureList.txt"));
            while((currentPictureName=reader.readLine())!=null)
{
ConsoleControl.systemPictureList.addElement(currentPictureName)
if(currentPictureName.contains("CRW")) continue;
        lastPicture=currentPictureName;
}
currentPictureShown = lastPicture;
currentPictureShownIndex = systemPictureList.getSize()-1;
            System.out.println(currentPictureShown);
            reader.close();
```

Metoda executePerlFetchFileList poziva Perl skriptu fetchFileList i postavlja varijable lastPictureTaken i cameraPictureList prema zapisu u datotekama lastPicture.txt i fileListParsed.txt koju kreira Perl skripta. Poziv Perl skripte obavlja se naredbom:

```
Runtime.getRuntime().exec("perl fetchFileList.pl")
```

Kod koji će dodati imena datoteka u listu cameraPictureList nalazi se u nastavku.

```
BufferedReader reader = new BufferedReader(new
FileReader("fileListParsed.txt"));
String currentPicture = "";
while((currentPicture=reader.readLine())!=null) {
ConsoleControl.cameraPictureList.addElement(currentPicture);
}
reader.close();
```

Metoda executePerlShoot prima argumente duljine ekpozicije, broja fotografija i kontrolne zastavice i poziva Perl skriptu shoot s argumentima duljine ekspozicije, broja fotografija i broja zadnje fotografije iz varijable lastPictureTaken. Prije poziva Perl skripte, poziva se metoda incCurrentPictureNumber s argumentom 1, a nakon poziva Perl skripte, ukoliko je kontrolna zastavica postavljena, metoda se poziva s argumentom broja fotografija umanjenim za jedan. Kod kojim se to ostvaruje nalazi se u nastavku.

```
incCurrentPictureNumber(1);
ls= Runtime.getRuntime().exec("perl shoot.pl "+shutterspeed+"
"+String.valueOf(numberOfShots)+" "+lastPictureTakenNumber());
input = new BufferedReader(new
InputStreamReader(ls.getInputStream()));
if(flag==1) incCurrentPictureNumber(numberOfShots-1);
```

Razred LuaFileCreator

Razred LuaFileCreator sadrži varijable zoom, ssv, av, iso, focus, flash te metode setZoom, setSsv, setAv, setIso, setFocus, setFlash, setShootValues i createLua. Sve metode osim createLua kao povratnu vrijednost vraćaju istinu ako su uspješno izvedene.

Metoda setZoom postavlja varijablu zoom ako je zadana vrijednost u rasponu od 0 do 14.

Metoda setSsv postavlja varijablu ssv ako je zadana vrijednost veća od 0.00001. Ssv predstavlja ekspoziciju. Vrijednost koja se zapisuje u varijablu je cjelobrojna vrijednost koja se dobije zaokruživanjem rezultata dobivenog formulom (1).

$$Ekspozicija = -96 \cdot \log_2(Trajanje\ ekspozicije) \quad (1)$$

Metoda setAv postavlja varijablu av ako je zadana vrijednost u intervalu od 2.6 do 5.5. Av predstavlja širinu otvora blende. Vrijednost koja se zadaje dobija se prema formuli (2).

$$Otvor\ blende = 192 \cdot \log_2(Širina\ otvora\ blende) \quad (2)$$

Metoda setIso postavlja varijablu iso ako je zadana vrijednost u intervalu od 6 do 8000. Vrijednost koja se postavlja dobija se prema formuli (3).

$$Osjetljivost = 96 \cdot \log_2(ISO\ vrijednost) - 227 \quad (3)$$

Metoda setFocus postavlja varijablu focus ako je zadana vrijednost u intervalu od 0 do 65535.

Metoda setFlash postavlja varijablu flash ako je zadana vrijednost nula, jedan ili dva.

Metoda setShootValues poziva sve gore navedene metode. Ako se sve metode uspješno izvedu, poziva metodu createLua koja će kreirati Lua skriptu. Kod kojim se to postiže nalazi se u nastavku.

```
if(lua.setAv(av) && lua.setFlash(flash) && lua.setIso(iso) &&
   lua.setSsv(ssv) && lua.setFocus(distance) &&
lua.setZoom(zoom)) {
        lua.createLua(numberOfShots,interval);
        return true;
        }
return false;
```

Metoda createLua kreira Lua skriptu luaScript.lua koristeći metodu println iz razreda PrintWriter. Skripta se sprema u trenutni radni direktorij.

Testiranje aplikacije na računalu

Prije testiranja rada na Beagleboardu, testirat će se na računalu na kojem je razvijana. Operacijski sustav instaliran na računalo je Ubuntu 12.04, procesor je dvojezgreni Intelov i3 s radnim taktom do 2.17 GHz, radna memorija je kapaciteta 3 GB a tvrdi disk kapaciteta 300 GB. Računalo podržava USB 2.0 priključak.

Prepoznavanje uređaja nakon priključivanja traje manje od sekunde. Vrijeme od pokretanja aplikacije do prikaza traje 18 sekundi. Promjena prikazane fotografije na prošlu ili slijedeću traje 3 sekunde. Povećanje i smanjenje fotografije traje 4 sekunde. Promjena parametara fotografiranja odvija se bez većeg kašnjenja. Uz vrijeme ekspozicije od 0.25 sekundi, vrijeme između pokretanja fotografiranja i prikaza fotografije je 25 sekundi.

Tijekom izvođenja aplikacije prikazuju se informativne poruke u polju za ispis teksta ispod tipke Shoot. Primjećuje se da se tijekom fotografiranja ne ispisuje poruka „Shooting in progress“ zadana u programu.

Fotografiranje tri fotografije s razmakom između fotografiranja od četiri sekunde traje 51 sekundu. Sve fotografije su prebačene sa fotoaparata na računalo. Prikazana je zadnja fotografija.

Fotografiranje tri fotografije s razmakom između fotografiranja od dvije minute traje 5 minuta i 17 sekundi. Dok aplikacija čeka, fotografija se prebacuje na računalo. Primjećuje se da se prikaz fotografije dok aplikacija čeka na fotografiranje slijedeće ne mjenja. Nakon završetka fotografiranja prikazuje se zadnja fotografija. Sve fotografije su prebačene na računalo.

Rezultati testiranja pokazuju da je potrebno dodati osvježavanje grafičkog sučelja nakon pokretanja fotografiranja.

U nastavku slijede primjeri fotografija s navedenim parametrima fotografiranja.

Slika 10 prikazuje lavandu. Fotografirana je u zatvorenoj prostoriji na dnevnom svijetlu.

Slika 10: Primjer slike - lavanda

Ekspozicija: 0.1 sekunda

Otvor blende: 2.6 mm

ISO: 300

Udaljenost objekta: 300 mm

Optičko povećanje: 0

Bljeskalica: isključena

Slika 11 prikazuje noćno nebo. Fotografirano je u naselju s javnom rasvjetom.

Slika 11: Primjer slike - noćno nebo

Ekspozicija: 20 sekundi

Otvor blende: 2.6 mm

ISO: 200

Udaljenost objekta: 65535 mm

Optičko povećanje: 0

Bljeskalica: isključena

Slika 12 fotografirana je za vrijeme noći uz nisku razinu osvjetljenja od javne rasvjete.

Slika 12: Primjer slike - noćna fotografija

Ekspozicija: 12 sekundi

Otvor blende: 2.6 mm

ISO: 200

Udaljenost objekta: 2000 mm

Optičko povećanje: 0

Bljeskalica: isključena

Testiranje aplikacije na Beagleboardu

Testiranje na Beagleboardu nije prošlo uspješno zbog tehničkih problema vezanih uz napajanje uređaja.

Zaključak

Aplikacija za kontrolu digitalnog fotoaparata izvedena je pomoću programskog jezika Java i skriptnih jezika Perl i Lua. Vrijeme od pokretanja fotografiranja do prikaza fotografije ograničeno je brzinom prebacivanja podataka preko USB-a i brzinom izvođenja skripti na fotoaparatu. Brzina izvođenja Jave i Perla je zadovoljavajuća.

Testiranje na Beagleboardu nije prošlo uspješno zbog tehničkih problema. Nakon otklanjanja problema, očekuju se pozitivni rezultati izvođenja programa uz sporije vrijeme izvođenja zbog slabijih performansi Beagleboarda.

Sustav se može dodatno proširivati tako da podržava *bracketing*, snimanje videa, stream videa sa fotoaparata te da omogućuje ponovno pokretanje fotoaparata preko računala. Dodatno se može implementirati kreiranje histograma na računalu i zebra način rada koji upozorava korisnika na preeksponirane fotografije.

Naprednim opcijama CHDK-a na fotoaparatu Canon Powershot A570 IS može se postići veća kvaliteta fotografije nego koristeći automatske postavke fotoaparata. Dulje ekspozicije od tvornički ograničenih, uz manju osjetljivost na svjetlo i ručno podešavanje fokusa daju jasnije i oštrije fotografije. Koristeći aplikaciju mogu se izvoditi složenije funkcionalnosti fotografiranja poput timelapsa, fotografiranja noćnih fotografija s velikim vremenom ekpozicije ili fotografiranja brzih pokretnih objekata s vrlo malim vremenima ekspozicije za smanjenje zamućenja.

Literatura

[1] Opis PTP-a

http://www.imaging.org/ist/resources/standards/ptp-standards.cfm

[2] Upute za korištenje PTP-a uz CHDK

http://chdk.wikia.com/wiki/PTP_Extension

[3] Popis funkcija za Lua skripte

http://chdk.wikia.com/wiki/CHDK_Scripting_Cross_Reference_Page

[4] Opis programa ptpcam

http://libptp.sourceforge.net/README

[5] Dokumentacija programskog jezika Java 7

http://docs.oracle.com/javase/7/docs/

[6] Priručnik za skriptni jezik Lua

http://www.lua.org/manual/5.1/

[7] Prof. dr. sc. Zoran Kalafatić, prof. dr. sc. Siniša Šegvić, doc. dr. sc. Stjepan Groš: Uvod u programski jezik perl, 02.04.2013

http://fer.unizg.hr/_download/repository/Skriptni_2_Perl%5B1%5D.pdf

[8] Specifikacije Canon Powershot A570 IS fotoaparata

http://www.cameras.co.uk/reviews/canon-powershot-a570-is.cfm

Naslov

Upravljanje digitalnim fotoaparatom pomoću protokola za prijenos fotografija

Sažetak

Cilj rada je pomoću protokola za prijenos fotografija ostvariti upravljanje fotoaparatom Canon Powershot A570 IS na operacijskom sustavu Linux. Upravljanje parametrima fotoaparata će se obavljati preko grafičkog sučelja. Fotografija će se prebaciti na upravljačko računalo i prikazati na ekranu. Izvođenje rješenja zadatka će se testirati na ugradbenom računalnom sustavu Beagleboard.

Na fotoaparat je postavljen CHDK *firmware* zbog potpune podrške za PTP. Za komunikaciju s fotoaparatom korišten je program ptpcam. Za razvoj aplikacije korišteno je NetBeans razvojno okruženje s programskim jezikom Java verzije 7. Za komunikaciju između ptpcama i Java programa korišten je skriptni jezik Perl verzije 5. Za izvođenje skripti na fotoaparatu korišten je skriptni jezik Lua verzije 5.1.

Rezultat rada je aplikacija s grafičkim sučeljem u kojoj se zadaju parametri fotografiranja i gdje se prikazuje fotografija. Testiranje aplikacije na Beagleboardu nije obavljeno. Zbog prenosivosti Java koda na druge operacijske sustave i podrške za skriptni jezik Perl na operacijskom sustavu Ångström očekuju se pozivitni rezultati sa sporijim vremenom izvođenja zbog slabijih performansi Beagleboarda u usporedbi s laptopima i osobnim računalima.

Ključne riječi

Protokol za prijenos fotografija, PTP, Canon Hack Development Kit, CHDK, ptpcam, Canon Powershot A570 IS, NetBeans, Java, Perl, Lua, Beagleboard

Title

Digital camera control using picture transfer protocol

Abstract

The goal of this paper is to control digital camera Canon Powershot A570 IS on Linux based operating systems using picture transfer protocol. Changing camera parameters will be done over graphical user interface. Photography will be downloaded to host computer and shown on screen. Results will be tested on embedded computer system Beagleboard.

The new firmware version has been uploaded to camera for full PTP support. Ptpcam has been used for communication between computer and camera. NetBeans development environment with Java 7 programming language has been used to create application with graphical user interface and for communication with Perl scripts. Perl v5 scripts have been used for communication between Java and ptpcam. Lua v5.1 scripts have been used for executing commands on camera.

The result of this paper is application with graphical user interface where users can change camera parameters and see photographs. Testing the application on Beagleboard hasn't been done. Positive test results are expected because of Java code portability on many operating systems and support for Perl scripting language on operating system Ångström which will be running Beagleboard. Also, because of Beagleboards less performance compared to laptops and personal computers, application execution is expected to be slower.

Keywords

Picture transfer protocol, PTP, Canon Hack Development Kit, CHDK, ptpcam, Canon Powershot A570 IS, NetBeans, Java, Perl, Lua, Beagleboard

Dodatak A

Perl skripta fetchFileList

```
#!perl

use List::Util qw(max);

printf("\n  Fetching file list. \n");
my $modeResponse = `./ptpcam --chdk="mode 0"`;
sleep(1);
my @fileListResponse = `./ptpcam --chdk="luar require
('lptpgui').dcimdl(false)"`;
sleep 3;
my @downLoadResponse = `./ptpcam --chdk="download A/ptpgui.txt
fileList.txt"`;
printf("\n Fetching file list succesfull. \n");
print("\n Searching for last picture. \n");
open FH, "<fileList.txt" or die $!; my @fileList=<FH>; close
FH;
my @pictureNames;
foreach $item (@fileList) {
   ($path,$name,$datetime,$junk)=split(/\|/,$item);
   if(index($path,"101CANON")!=-1) {
      push(@pictureNames,$name);
   }
}
#search for max
my @array2;
my @finalarray;
foreach $item (@pictureNames) {
   @itemsplit=split(/\./,$item);

   foreach $peace (@itemsplit) {
      if(index($peace,"IMG_") != -1) {
         #$peace contains all images in format IMG_xxxx
         @pictureNumbers=split(/\x5F/,$peace); #split by _
         foreach $number (@pictureNumbers) {
         if(index($number,"IMG") == -1) {
               push(@finalarray,$number);
            }
         }

      }
```

```
   }
}
$high=max(@finalarray);
$lastPictureName="IMG_".$high.".JPG";
printf("$high\n");


open($myfile,'>lastPicture.txt');
print $myfile $lastPictureName;
close($myfile);
print("\n Last picture = $high. All done. \n");
exit(0);
```

Dodatak B

Perl skripta fetchLocalPictureList

```
#!perl

printf("\n  Fetching picture list. \n");
$returnMessage = `ls`;
@returnMessageArray=split(/(\s)/,$returnMessage);
open($myfile,'>pictureList.txt');
foreach $file (@returnMessageArray) {
   if((index($file,"IMG")!=-1)||(index($file,"CRW")!=-1)) {
   print $myfile $file;
   print $myfile "\n";
   }
}
close($myfile);
printf("\n  Done \n");
```

Dodatak C

Perl skripta commands

```
#!perl

use feature qw(switch);

my ($command,$filename) = @ARGV;
given($command){
  when(1) {
    printf("\n Backlight ON \n");
    my @response = `./ptpcam --chdk="lua set_backlight(1)"`;
  }
  when(2) {
    printf("\n Backlight OFF \n");
    my @response = `./ptpcam --chdk="lua set_backlight(0)"`;
  }
  when(3) {
    printf("\n RAW enabled \n");
    my @response = `./ptpcam --chdk="lua set_raw(1)"`;
  }
  when(4) {
    printf("\n RAW disabled \n");
    my @response = `./ptpcam --chdk="lua set_raw(0)"`;
  }
  when(5) {
    printf("\n RAW noise reduction ON \n");
    my @response = `./ptpcam --chdk="lua set_raw_nr(1)"`;
  }
  when(6) {
    printf("\n RAW noise reduction OFF \n");
    my @response = `./ptpcam --chdk="lua set_raw_nr(0)"`;
  }
  when(7) {
    printf("\n Reading temperature \n");
    open($myFile,'>temperature.txt');
    my @response = `./ptpcam --chdk="luar get_temperature(0)"`;
    printf("\n@response\n");
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
        print $myFile $responseSplit2;
        print $myFile "\n";
```

```
      }
    }
    my @response = `./ptpcam --chdk="luar get_temperature(1)"`;
    printf("\n@response\n");
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
        print $myFile $responseSplit2;
        print $myFile "\n";
      }
    }
    my @response = `./ptpcam --chdk="luar get_temperature(2)"`;
    printf("\n@response\n");
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
        print $myFile $responseSplit2;
        print $myFile "\n";
      }
    }
    close($myFile);
  }
  when(8) {
    open($myFile,'>SDcard.txt');
    printf("\n SD card \n");
    my @response = `./ptpcam --chdk="luar get_disk_size()"`;
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
        print $myFile $responseSplit2;
        print $myFile "\n";
      }
    }
    printf("\n@response\n");
    my @response = `./ptpcam --chdk="luar
get_free_disk_space()"`;
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
```

```
        print $myFile $responseSplit2;
        print $myFile "\n";
      }
    }
    printf("\n@response\n");
    my @response = `./ptpcam --chdk="luar get_jpg_count()"`;
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
        print $myFile $responseSplit2;
        print $myFile "\n";
      }
    }
    printf("\n@response\n");
    my @response = `./ptpcam --chdk="luar get_raw_count()"`;
    foreach $item (@response) {
      if(index($item,"ret:")!=-1) {
        $responseSplit=(split(' ',$item))[0];
        $responseSplit2=(split(':',$responseSplit))[2];
        printf("\n$responseSplit2\n");
        print $myFile $responseSplit2;
        print $myFile "\n";
      }
    }
    printf("\n@response\n");
    close($myFile);
  }
  when(9) {
    printf("\n Delete file $filename on system \n");
    my @response = `rm $filename`;
  }
  when(10) {
    printf("\n Delete file $filename on camera \n");
    $cmd="./ptpcam --chdk=\"luar
os.remove(\\\"A/DCIM/101CANON/$filename\\\")\"";
    printf("\n$cmd\n");
    my @response =`$cmd`;
    printf("\n@response\n");
  }
  default { printf("\nError on command argument\n"); }
}
```

Dodatak D

Perl skripta shoot

```
#!perl
 #-------------------------------------------------------------------
-------------
 # Optionally upload/run script to/on camera
 #-------------------------------------------------------------------
-------------
 my $myScript="luaScript.lua";
 my $upload=1;
 my $run=1;
 my ($shutterspeed,$numFilesToGet,$pictureNumber,$raw) = @ARGV;
 #-------------------------------------------------------------------
-------------
#upload script
  if ($upload) {
   printf("\n  Uploading $myScript script to camera. \n");
   my @upLoadResponse = `./ptpcam --chdk="upload $myScript
A/CHDK/SCRIPTS/$myScript"`;
   }
#run script
 if ($run) {
   if ($upload) { sleep 3; }
   $scriptMessage = `./ptpcam --chdk="getm"`; #discard existing
messages
   sleep 1;
   printf("\n  Running $myScript script on camera.\n");
   my $modeResponse = `./ptpcam --chdk="mode 1"`;
   sleep 3;
   my $scriptResponse = `./ptpcam --chdk="lua
require('lptpgui').exec_luafile([[A/CHDK/SCRIPTS/$myScript]])"`
;
   printf("  > launch response= $scriptResponse\n");
   my $wait=100;
   while ($wait) {
     sleep 3;
     $scriptStatus = `./ptpcam --chdk="script-status"`;
     ($junk,$run,$msg)=split(/\s+/,$scriptStatus);
     printf("  > status: $run - $msg\n");
     $wait--; if ($run =~ /no/) {$wait=0;}
     }
   sleep $shutterspeed;
```

```
   if ($msg =~ /yes/) {
     sleep 1;
     $scriptMessage = `./ptpcam --chdk="getm"`;
     printf("\n  Output From Script PRINT
command(s):\n\n$scriptMessage\n  --\n");
     }
   my $modeResponse = `./ptpcam --chdk="mode 0"`;
 }
   sleep 1;
   #download pictures

   for ($i=$pictureNumber;$i<=($pictureNumber+$numFilesToGet-
1);$i++) {
     printf("fetching IMG_$i \n");
     $result=`./ptpcam --chdk="download
A/DCIM/101CANON/IMG_$i.JPG IMG_$i.JPG"`;
     sleep 2;
     if ($raw==1) {
       $result=`./ptpcam --chdk="download
A/DCIM/101CANON/CRW_$i.DNG CRW_$i.DNG"`;
       sleep 2;
     }
   }

 printf("\n  All done.\n"); exit(0);
```

Dodatak E

Primjer kreirane Lua skripte

```
--[[
@title AdvancedRemoteCapture
--]]
loop=3
interval=5
counter=interval
zoom=1
ssv=31
av=295
iso=603
focus=400
flash=2
while loop > 0 do
    set_zoom(zoom)
    sleep(1000)
    set_tv96_direct(ssv)
    sleep(500)
    set_av96(av)
    sleep(500)
    set_sv96(iso)
    sleep(500)
    set_focus(focus)
    sleep(500)
    set_prop(143,flash)
    sleep(500)
    press("shoot_half")
    sleep(1000)
    press("shoot_full")
    sleep(2000)
    release("shoot_full")
    release("shoot_half")
    loop=loop-1
    if loop > 0 then
        counter=interval
        while counter > 0 do
            counter = counter -1
            sleep(1000)
        end
    end
end
```