Algoritam za provjeru sinkronizacije video sadržaja i podnaslova

Kedačić, Davor

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni diplomski studij

ALGORITAM ZA PROVJERU SINKRONIZACIJE VIDEO SADRŽAJA I PODNASLOVA

Diplomski rad

Davor Kedačić

Osijek, 2017.

Sadržaj

1. UVOD

Tema ovog diplomskog rada dobivena je i odrađena u suradnji s institutom RT-RK za potrebe testiranja HbbTV (*engl. Hybrid Broadcast Broadband TV*) aplikacije. Tema se bavi provjerom ispravnosti sinkronizacije video sadržaja s podnaslovima (*eng. subtitle*) unutar HbbTV standarda. HbbTV je industrijski standard koji omogućuje proširenje standardnih mogućnosti koje posjeduje digitalna televizija s dodatnim interaktivnim aplikacijama koje su bazirane na web tehnologijama. Kako se digitalna televizija razvijala također se javila potreba za novim tipom podnaslov datoteka koje bi se koristilo unutar navedenog HbbTV standarda. Taj novi tip podnaslov datoteka je definiran EBU-TT-D standardnom. Rad je izrađen u programskom jeziku C uz dodatak izrade skripte za generiranje testnih sekvenci. Skripta se koristi za potrebe provjere ispravnosti algoritma i izrađena je u programsku jeziku JavaScript koristeći Node.js radno okružje. U nastavku rada će biti definirane i detaljno opisane sve tehnologije korištene za izradu algoritma. Također će biti opisana problematika zadatka i na kraju rada će biti predstavljeni rezultati provedenih testiranja.

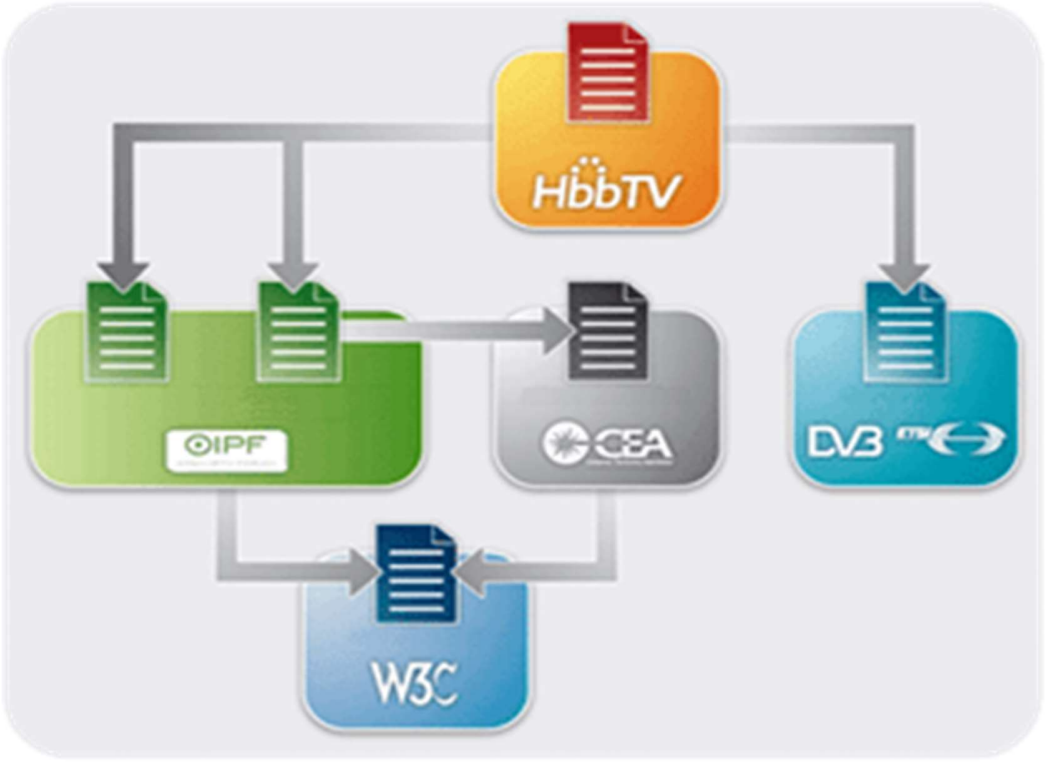
1.1 Zadatak diplomskog rada

Kod prikaza podnaslova nad video sadržajima u HbbTV okruženju, postavlja se zahtjev da točnost pojavljivanja podnaslova mora biti na razini okvira. HbbTV specifikacija definira specijalizirane funkcije za provjeru sinkronizacije video sadržaja s podnaslova, pri čemu se sama provjera oslanja na posebne hardverske komponente. Cilj ovog rada je razvoj softverskog rješenja koje bi funkcioniralo u skladu sa specifikacijom, a bilo bi u stanju provjeru sinkronizacije obaviti izvan realnog vremena na osnovu video snimke izvršenja testa na uređaju. Video snimak može poticati s kamere ili hvatača (*eng. grabber*) uređaja.

2. TEORIJSKA PODLOGA

2.1 HbbTV

HbbTV je globalna inicijativa kojoj je cilj razviti industrijski standard za razvoj i rad s hibridnom i interaktivnom digitalnom televizijom. Usmjerena je na usklađivanje emitiranja i širokopojasnog pružanja zabavnih usluga potrošačima putem povezanih televizora, upravljačke kutija za televiziju (*eng. set-top box*) i uređaja s više zaslona. HbbTV specifikacije razvile su vodeće tvrtke u televizijskoj industriji radi poboljšanje korisničkog doživljaja omogućavanjem pristupa inovativnim i interaktivnim uslugama. Specifikacije na kojima se temelji koriste elemente postojećih specifikacije iz drugih standarda, uključujući OIPF (*eng. Open IPTV Forum*), CEA (*eng. Consumer Electronics Association*), DVB (*eng. Digital Video Broadcasting*), MPEG-DASH (*eng. Dynamic Adaptive Streaming over HTTP*) i W3C (*eng. World Wide Web Consortium*) (Sl. 2.1) [1].



Slika 2.1 HbbTV specifikacija i odnos prema drugim specifikacijama[1]

Koristeći postojeće specifikacije iz drugih standarda pokušava se stvoriti standard za isporuku usluga do korisnika putem jednog korisničkog sučelja te time stvoriti otvorenu platformu kao alternativu zatvorenim platformama čime bi se omogućio brži razvoj novih inovativnih aplikacija i efikasnije korištenje postojećih sadržaja. Proizvodi i usluge koji koriste HbbTV standard mogu raditi na različitim tehnologijama emitiranja, kao što su satelitske, kabelske ili zemaljske mreže. Usluge koje HbbTV omogućuje su:

- Elektronski programski vodič (*eng. Electronic Program Guide* - EPG)
- Video na zahtjev (*eng. Video On Demand* VOD)
- Napredni teletekst
- Pristup društvenim mrežama
- Interaktivne igre i multimedijske aplikacije
- Interaktivni oglasi
- Glasanje

2.2 Korištene tehnologije

2.2.1 C programski jezik

C programski jezik je jezik opće namjene koji omogućuje kompleksne zapise podataka (strukture, pokazivači, nizovi, liste i drugi izrazi), modernu kontrolu programskog toka i podatkovnih struktura te bogat set operatora. C programski jezik je jezik niske razine koji nije specijaliziran za posebno područje uporabe. Navedeno je i jedna od njegovih najjačih strana jer mu omogućuje raznovrsnu uporabu. Dennis Richie, tvorac C programskog jezika, ga je originalno razvio za UNIX operativni sustav. Operacijski sustav, C prevodilac i gotovo sve UNIX aplikacije su napisane u C programskom jeziku. C programski jezik nije vezan za nikakvo posebno sklopovlje ili sistem stoga je lagano pisati programe koji će se izvoditi bez promjena na bilo kojem uređaju koji podržava C programski jezik [2].

2.2.2 Node.js

Node.js je višeplatformsko JavaScript okružje otvorenog koda za izvođenje JavaScript koda na strani poslužitelja. JavaScript se prvenstveno koristio za skriptiranje na strani klijenta, gdje su skripte napisane u JavaScriptu ugrađene u HTML (*eng. HyperText Markup Language*) web stranice, koje se pokreću na strani klijenta pomoću JavaScript okružja u korisničkom web pregledniku. Node.js omogućuje JavaScriptu da se koristi za skriptiranje na strani poslužitelja, pokretanje skripti

na strani poslužitelja te aktiviranje dinamičkog sadržaj web stranice prije nego što se stranica prikaže na korisnikovom web pregledniku. Koristeći ovaj način rada Node.js omogućuje razvoju web aplikacija ujedinjenje oko jednog programskog jezika, za razliku od oslanjanja na drugi jezik za pisanje skripti na strani poslužitelja [3]. Kako će se skripta za izradu testnih sekvenci izrađena u ovom radu pokretati na klijentskoj strani mogla je biti izrađena u čistom JavaScriptu ali je Node.js korišten zbog jedne svoje ekstenzije. To je NPM (*eng. Node Package Manager* - NPM) ekstenzija koja omogućuje pristup gotovim bibliotekama, u slučaju ovog rada konkretno biblioteci za generiranje koda brzog odgovora (*eng. Quick Response code* – QR kod) i biblioteci za izradu XML (*eng. eXtensible Markup Language)* dokumenata.

NPM registar sadrži gotovo pola milijuna paketa besplatnog Node.js koda, što ga čini najvećim registrom softvera na svijetu. U osnovi NPM paketi su već gotovi programski moduli otvorenog koda izrađeni u Node.js-u.

2.2.3 XML

XML je označni jezik koji definira skup pravila za kodiranje dokumenata u obliku koji je čitljiv ljudima i računalima. W3C XML specifikacije i nekoliko drugih srodnih specifikacija od kojih su sve slobodni otvoreni standardi definiraju XML.

Cilj XML-a je jednostavnost, općenitost i upotrebljivost preko Interneta. To je tekstualni format podataka koji koristi Unicode (*eng. UNIversal trunk - out-of-service CODE*) set znakova kako bi pružio podršku za različite ljudske jezike. Unicode je međunarodni standard kodiranja za upotrebu s različitim jezicima i skriptama, pri čemu svakom slovu, znamenki ili simbolu dodjeljuje jedinstvenu numeričku vrijednost koja se primjenjuje na različitim platformama i programima.

Iako je dizajn XML-a usredotočen na dokumente, jezik se široko koristi za prikazivanje proizvoljnih struktura podataka kao što su one koji se koriste u web uslugama.

Postoji nekoliko shema koje pomažu pri definiranju dokumenata koji se temelje na XML-u, a programeri su razvili i mnoga sučelja aplikacijskog programiranja (*eng. Application Programming Interface* - API) za pomoć pri obradi XML dokumenata [4].

2.2.4 FFmpeg

FFmpeg (*eng. Fast Forward Moving Pictures Expert Group*) je besplatni softverski projekt koji kreira biblioteke i programe za rukovanje multimedijskim podacima. *FFmpeg* uključuje *libavcodec*, biblioteku audio/video kodeka, *libavformat* (*Lavf*), audio / video kontejnersku mux i

demux biblioteku, te program *ffmpeg* za transkodiranje multimedijskih datoteka. *FFmpeg* je izdan pod *GNU Lesser General Public License 2.1+* ili *GNU General Public License 2+* (ovisno o tome koje opcije su omogućene) [5].

2.2.5 QR kod

QR kod je dvodimenzionalni (2D) matrični kod koji spada u porodicu kodova koji su čitljivi strojevima (Sl. 2.2). Svi kodovi u toj porodici se nazivaju bar kodovi (*eng. barcode)* neovisno o tome da li se sastoje od linija, kocki ili drugih elemenata raznih oblika. U usporedbi s 1D kodovima, 2D kodovi mogu sadržavati puno više informacija unutar manjeg prostora, a QR kod, u usporedbi s drugim 2D kodovima, može sadržavati još više informacija. QR kod također posjeduje napredne metode korekcije pogrešaka i ostale jedinstvene karakteristike koje ga čine bržim i pouzdanijom od ostalih kodova.

Slika 2.2 QR kod

Kao i pisani jezik, QR kodovi su vizualna reprezentacija informacije, ali za razliku od pisanog jezika čitljivog ljudima, QR kodovi su dizajni da bi bili čitljivi i razumljivi to jest mogu biti dekodirani od strane računala. Računala ih dekodiraju koristeći sustave za strojni vid koji se sastoje od optičkih laserskih skenera ili kamera i programa koji dekodiraju QR kodove. Pravila stvaranja QR kodova, to jest njena „gramatika“ i setovi znakova koji mogu biti upisani u njih, nazivaju se simbologija QR kodova [6].

Za razliku od 1D kodova, QR kod je 2D matrični kod koji prijenosi podatke ne veličinom i položajem crta te razmaka između njima u horizontalom smjeru već koristeći tamne i svijetle

elementa koji se nazivaju moduli reprezentiranim u stupcima i redovima u horizontalnom i vertikalnom smjeru (Sl. 2.3).

Slika 2.3 Razlika u pohrani podatka 1D bar koda i QR koda

Svaki od tamnih ili svijetlih elemenata QR koda predstavlja 0 ili 1 i iz toga razloga je čitljiv strojevima. Svi QR moduli obavljaju neku funkciju, neki sadržavaju informacije, dok su drugi grupirani u određene funkcionalne uzorke koji povećavaju čitljivost i omogućavaju ispravke pogrešaka te kompenzaciju distorzije. Sam oblik koda omogućava programima za dekodiranje da znaju koje je veličine QR kod. Također postavlja se zahtijeva za postojanjem zone oko QR koda u kojoj ne smiju biti nikakve informacije i koji mora biti širok 4 elementa kako se okolni tekst ili oznake ne bi interpretirali kao dio koda.

Kod drugih 2D matričnih kodova potrebno je mnogo vremena kako bi se utvrdila orijentacija koda, njegov položaj izražen u x i y koordinatama i veličina. Kako bi riješio ovaj problem QR kod je dizajniran s posebnim uzorcima za određivanje položaja koji se nalaza u gornjem lijevom, gornjem desnom i donjem lijevom kutu. Uzorci su simetrično postavljeni u odnosu 1:1:3:1:1 što im omogućuje da se mogu skenirati iz bile kojeg položaja unutar 360 stupnjeva. Kao rezultat toga QR kod ima i do 20 puta brže vrijeme čitanja nego ostali matrični kodovi. Prikaz građe QR koda je prikazan na slici 2.4.

Slika 2.4 Moduli QR koda

QR kod može biti generiran u 40 različitih verzija koji se razlikuju po broju modula od kojih se sastoje. Verzija 1 se sastoji od 21x21 modula, verzija 2 od 25x25 modula dok se verzija 40 sastoji od 177x177 modula (Sl. 2.5) . Svaki od modula sadržava još četiri dodatna modula sa svake strane u kojim se ne nalaze podaci što je 16 dodatnih modula po kodu. Količina podataka koje QR kod može sadržavati je određen njegovom verzijom, tipom znakova koji su u njega upisani i razinom korekcije grešaka [7].

Slika 2.5 Verzije QR kodova

3. ZAHTJEVI SUSTAVA

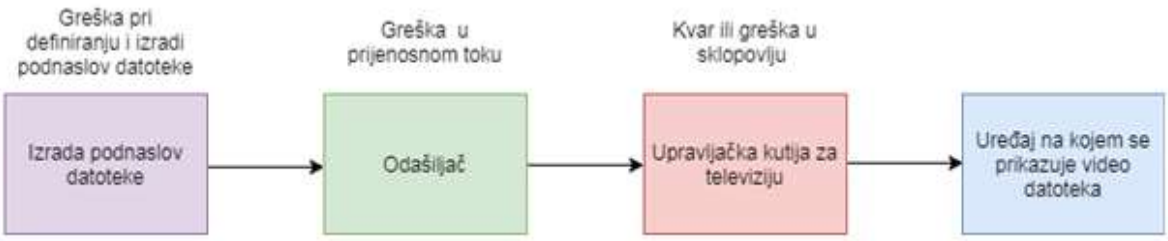
3.1 Detekcija tipa sinkronizacije

U sklopu zadatka je potrebno razviti algoritam koji provodi detekciju točnosti pojavljivanja početka i kraja podnaslova u video datoteci na razini video okvira. Potrebno je analizirati vremena pojavljivanja podnaslova na zaslonu korisnika te na osnovu tih vremena analizirati točnost pojavljivanja. Razine točnosti možemo definirati kao sljedeća stanja:

1. Podnaslov je počeo u točno zadanom vremenu početka
2. Podnaslov je završio u točno zadanom vremenu završetka
3. Podnaslov je počeo prije zadanog vremena početka
4. Podnaslov je počeo nakon zadanog vremena početka
5. Podnaslov se završio prije zadanog vremena završetka
6. Podnaslov se završio poslije zadanog vremena završetka
7. Podnaslov se uopće nije pojavio na zaslonu

Bio bi velik problem ručno detektirati nepravilnosti u sinkronizaciji video datoteke i pratećeg podnaslova, jer bi to zahtijevalo da netko fizički prati promjene podnaslova na zaslonu te vizualno detektira da li se taj podnaslov pojavio u pravom trenutku, stvara se potreba za izradom programskog algoritma koji će riješiti taj problem, to jest detektirati sve nepravilnosti u sinkronizaciji. Koristeći taj algoritam konkretno je potrebno odrediti točnu kombinaciju stanja u kojoj se nalazi analizirana podnaslov datoteka kako bi se eventualne greške mogle naknadno ispraviti.

Razloge zbog kojih se može dogoditi da podnaslovi i video sadržaj nisu sinkronizirani možemo podijeliti u tri šire kategorije. Te kategorije mogu biti uzrokovane situacijom u kojoj se greška u sinkronizaciji razvila još u stadiju kad je podnaslov datoteka izrađivana. Razlog toga može biti pogrešna definicija ili pogrešno uneseno početno i krajnje vremena pojave podnaslova. Također moguće su greške u prijenosnom toku koji prenosi informacije o podnaslovu te kvar same upravljačke kutije za televizijski prijenos što rezultira krivim prikazom podnaslova na zaslonu. Grafički prikaz kategorija je dan na slici 3.1.

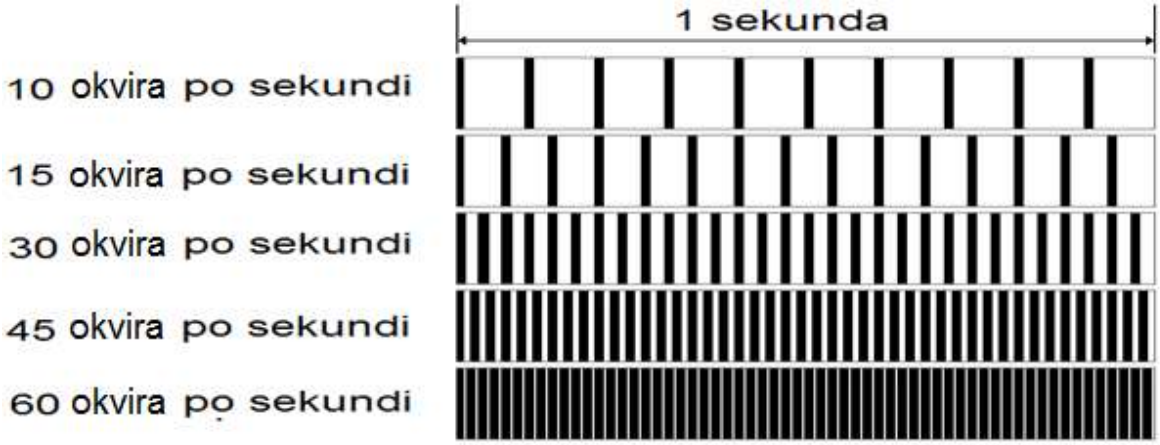


Slika 3.1 Mogući izvori neusklađenosti podnaslova i video sadržaja

Kako se zahtjev ovog zadatka odnosi samo na detekciju sinkronizacije na krajnjoj točki prijenosa, odnosno obrađuje se snimljena video datoteka, rezultat rada algoritma treba ukazivati samo jesu li podnaslov i video sadržaj u sinkronizaciji to jest ne treba ukazati na kojem dijelu se dogodila greška u sinkronizaciji.

3.2 Način identifikacije trenutnog video okvira

Kako video datoteka može biti izrađena snimajući sadržaj koji potječe s uređaja za hvatanje sadržaja ili snimajući zaslon kamerom potrebno je osmisliti način identifikacije rednog broja video okvira (*eng. frame*) kako bi se moglo znati na kojem video okviru se pojavio i završio podnaslov. Iako je potrebno analizirati svaki video okvir jedan za drugim, ne može se osloniti na broj video okvira u video datoteci. Razlog toga su razlike u broju okvira video datoteke koje nastaju kada se video sadržaj koji se prikazuje na zaslonu uređaja za reprodukciju snima kamerom u većem broju okvira po sekundi (*eng. frame rate*) od broja okvira po sekundi izvorne video datoteke.



Slika 3.2 Grafički prikaz broja okvira po sekundi

Na slici 3.2 je prikazano kako se ponaša video s promjenom broja okvira po sekundi, jasno je vidljivo kakvi problemi mogu nastati prilikom snimanja videa sadržaja koji je kodiran u *n* okvira po sekundi kamerom koja snima u *m* okvira po sekundi. Ako se snima video koji je izvorno kodiran u 30 okvira po sekundi kamerom koja snima u 60 okvira po sekundi dobit će se video datoteka koja se sastoji od više video okvira nego što ih ima izvorni video koji se snimao. Obrnuti proces je istinit i za video koji je izvorno kodiran u 60 okvira po sekundi a snima se kamerom u 30 okvira po sekundi. Video nastao takvim postupkom će se sastojati od manjeg broja video okvira. Kada se video kodiran u 30, to jest 60 okvira po sekundi snima s kamerom u 30, to jest 60 okvira po sekundi, krajnja video datoteka bi se trebala sastojati od aproksimativno identičnog broja okvira.

Kako snimanje ne može „stvoriti“ nove video okvire, u slučaju snimanja kamerom koja snima u većem broju okvira po sekundi nego što ih ima video koji se snima neki video okviri će se ponoviti dva ili više puta kako bi se postigao zadovoljavajući broj okvira po sekundi novog videa.

Količina novih video okvira koji će nastati ovisi o raznim ograničenjima u kompresiji i kodiranju video datoteka.

Ti novonastali video okviri onemogućuju korištenje brojanja video okvira kako bi se provjerilo da li se podnaslov pojavio ili završio na pravom video okviru. Razlog toga leži u tome što podnaslov datoteka pisana za video datoteku kodiranu u 30 okvira po sekundi ima definirano početno vrijeme podnaslova na *n* video okviru. Ako se taj video snima s kamerom koja snima u 60 okvira po sekundi te vrši usporedbu s zadanom podnaslov datotekom podnaslovi će se pojaviti u odmaku *m* zbog novonastalih video okviru. Samim time algoritam će kao rezultat registrirati da podnaslov i video datoteka nisu sinkronizirani iako su video datoteka i podnaslov datoteka ustvari sinkronizirani ali postoji odstupanje zbog većeg broja video okvira. Taj problem bi se mogao riješiti tako da se zna broj okvira po sekundi izvornog videa, a i novog videa nastalog snimanjem kamere i na osnovu toga izračuna odstupanje u broju video okvira, to jest koliko puta će se svaki video okvir ponoviti. Međutim, kako algoritam radi s gotovim video datotekama, nema informacije o broju okvira po sekundi video datoteke iz koje je nastala video datoteka s kojom se radi te stoga ovaj način praćenja video okvira nije pogodan.

Drugi način praćenja video okvir koji se obrađuje bi bio zapisivanje ili praćenje vremena u kojem se pojavljuje podnaslovu na video okviru. Ali takav način praćenja sa sobom nosi problem. Naime, u podnaslov datoteci može biti definirano da podnaslov počinje u vremenu *TT:TT:TT*

(*format sat:minuta:sekunda*) izvornog videa, ali pri snimanju kamerom snimanje nije počelo u vremenu 00:00:00 izvornog videa već je snimanje počelo nakon što je prošlo *t* sekundi izvornog videa. Problem će nastati kada algoritam pokuša analizirati da li se podnaslov stvarno pojavio u zadanom vremenu *T* koristeći snimanu video datoteku, te će rezultat biti negativan jer će pojavljivanje podnaslova biti pomaknuto za *T* - *t*, gdje *t* predstavlja vrijeme koje je proteklo u izvornom videu prije nego se kamerom počela snimati video sekvenca a *T* predstavlja vrijeme pojave u izvornom videu. Zbog toga razloga nije prigodno označavati video okvire koristeći proteklo vrijeme.

Iz svega navedenog jasno je da je jedan od zahtjeva ovoga zadatka naći način kojim se može pouzdano znati na kojem o koje video okviru se radi neovisno o broju video okvira ili vremena trajanja video datoteke koju analiziramo.

Zahtjev za identifikaciju trenutnog video okvira je riješen korištenjem QR koda. QR kod je pozicioniran u gornjem desnom kutu video okvira i udaljen je od ruba za 100 elemenata slike s lijeve strane i 100 elemenata slike s gornje strane (Sl. 3.3).

Tablica 3.3 Video okvir s pozicioniranim QR kodom

Tako je izbjegnuto bilo kakvo brojanje video okvira i utjecaj dodatnih video okvira nastalih snimanjem kamerom koja snima u većem broju okvira po sekundi od broja okvira po sekundi originalne video datoteke. Dodani broj video okvira ne stvara problem koristeći ovaj način označavanja jer su ti video okviri nastali ponavljanjem već postojećih video okvira tako da u slučaju da se *n* puta ponovi isti video okvir na svakom od njih će pisati isti QR kod.

QR kod je također korišten kako bi se smanjio utjecaj smanjene kvalitete video okvira koji nastanu snimanjem kamerom. Kako na kvalitetu video datoteke snimane kamerom utječu vanjski faktori kao što su osvjetljenje prostorije, razina osvjetljenja zaslona, kut snimanja te sama kvaliteta kamere, QR kod je bio pravo rješenje za označavanje video okvira jer on u sebi sadrži algoritme za poravnavanje i ispravljanje grešaka. Problemi s čitljivošću QR koda mogu nastati zbog lošeg kuta snimanja ili nedovoljne kvalitete video okvira koji se analizira.

Algoritam za poravnavanje omogućuje da se do neke mjere kompenzira utjecaj kuta snimanja na čitljivost QR koda kako bi se što preciznije mogao odrediti broj trenutnog video okvira.

Algoritam za ispravljanje grešaka je ugrađen u sam QR kod. Taj algoritam se bazira na tome da u QR kod dodaje Reed-Solomon kodove. Reed-Solomon kodovi su blokovski kodovi za ispravljanje pogrešaka koji se koriste u digitalnim komunikacijama [8]. Ovo omogućuje QR kodu da bude čitljiv čak i u slučaju lošije kvalitete video okvira s kojeg se dekodira QR kod. Postoje 4 nivoa ispravljanja grešaka. Što je razina korekcije veća to je i QR kod veći.

Razina korekcije greške	Postotak korekcije
L	7%
M	15%
Q	25%
H	30%

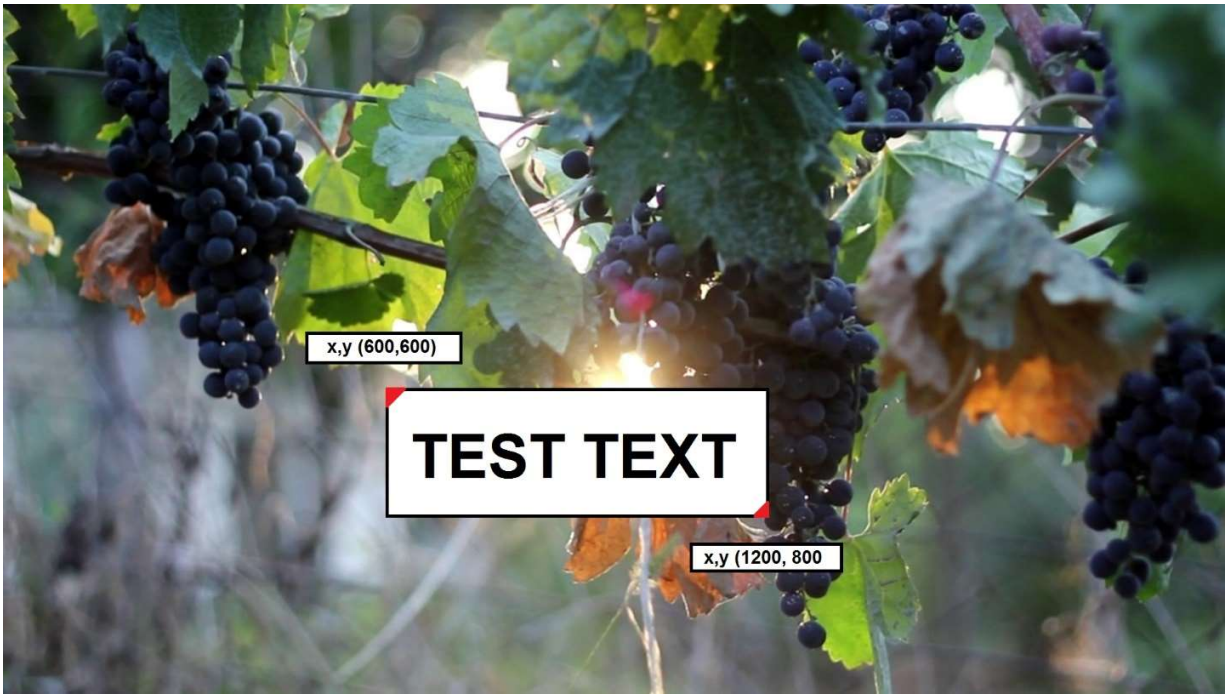
Tablica 3.1 Razine korekcije grešaka QR koda

Pri odabiru razine korekcije potrebno je uzeti u obzir okolinu u kojoj se koristi. Razine Q i H se koriste u okružjima u kojim se QR kod može oštetiti ili biti nečitljiv zbog prljavštine i drugih vanjskih faktora. U slučaju ovog zadataka zna se da će se koristiti u kontroliranim uvjetima tako da

se koristi QR verzija 1 koja se sastoji od 21x21 modula, jer su QR kodu zapisana mala količina podatka (broj video okvira), te razina korekcije greške M (15 %) što daje optimalnu točnost i brzinu dekodiranja.

3.4 Način održavanjem položaja elemenata video okvira

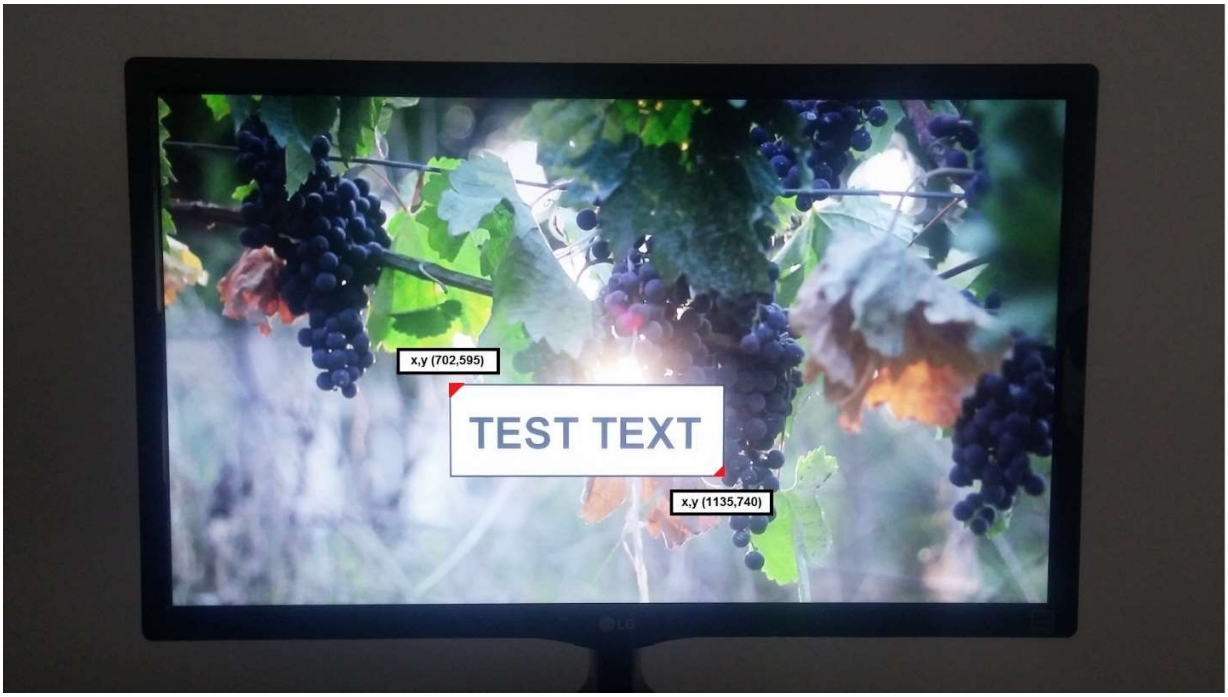
Jedan od zahtjeva zadatka je snimanje vršiti na način da se dobiju što dosljedniji rezultati kada se snimanje zaslona vrši kamerom. Udaljenost kamere od zaslona te kut snimanja mogu utjecati na kvalitetu snimljene video datoteke i relativan položaj elemenata na ekranu. Taj problem je primjetan ako se za primjer uzme jedan video okvir video datoteke koji ima rezoluciju 1920x1080 elemenata slike i koji je snimana uređajem za hvatanje sadržaja na kojoj se nalazi neki elemenat na zadanim koordinatama prikazan na slici 3.4.



Slika 3.4 Video okvir snimljen uređajem za hvatanje sadržaja

Nakon toga se uzme isti taj okvir iste video datoteke snimane kamerom koja je također u rezoluciji od 1920x1080 elemenata slike, ali je snimanje obavljeno s 20 cm udaljenosti od zaslona prikazano na slici 3.5. U tom slučaju elementi na video okviru snimane video datoteke neće imati iste *x* i *y* koordinate na razini video okvira kao i elementi video datoteke s koje je snimano iz razloga što je snimanje rađeno s određene udaljenosti od ekrana. To dovodi do toga da sama slika koju treba

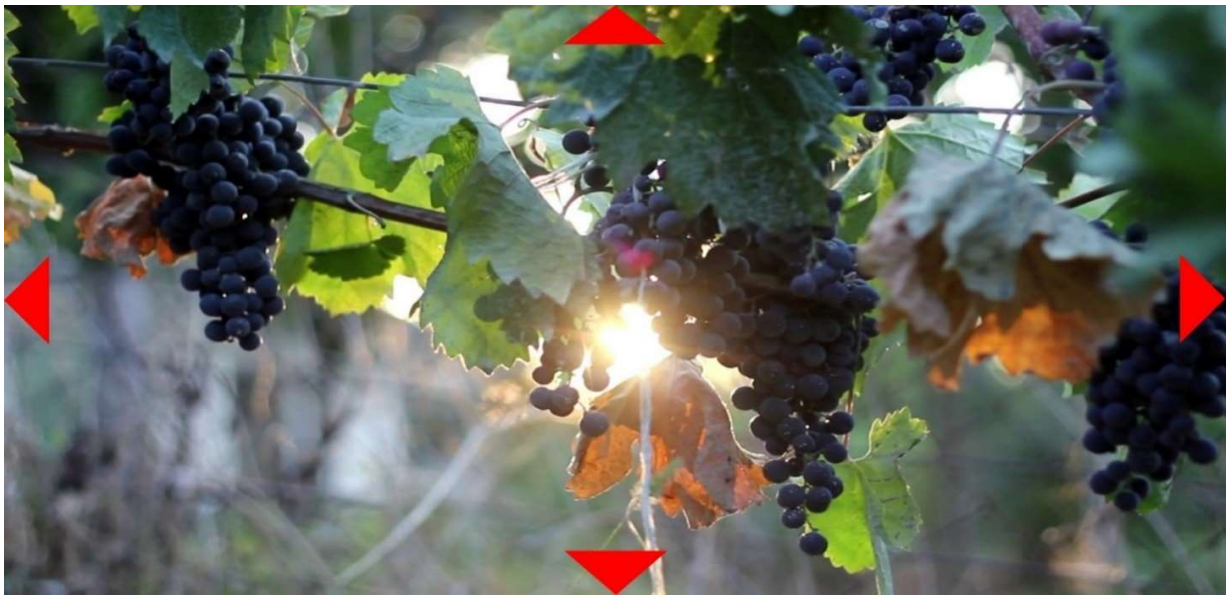
obraditi više nije u fokusu već se sa strane pojavljuju elementi okoline uzrokovane udaljavanjem kamere.



Slika 3.5 Video okvir snimljen kamerom s udaljenosti od 20 centimetara

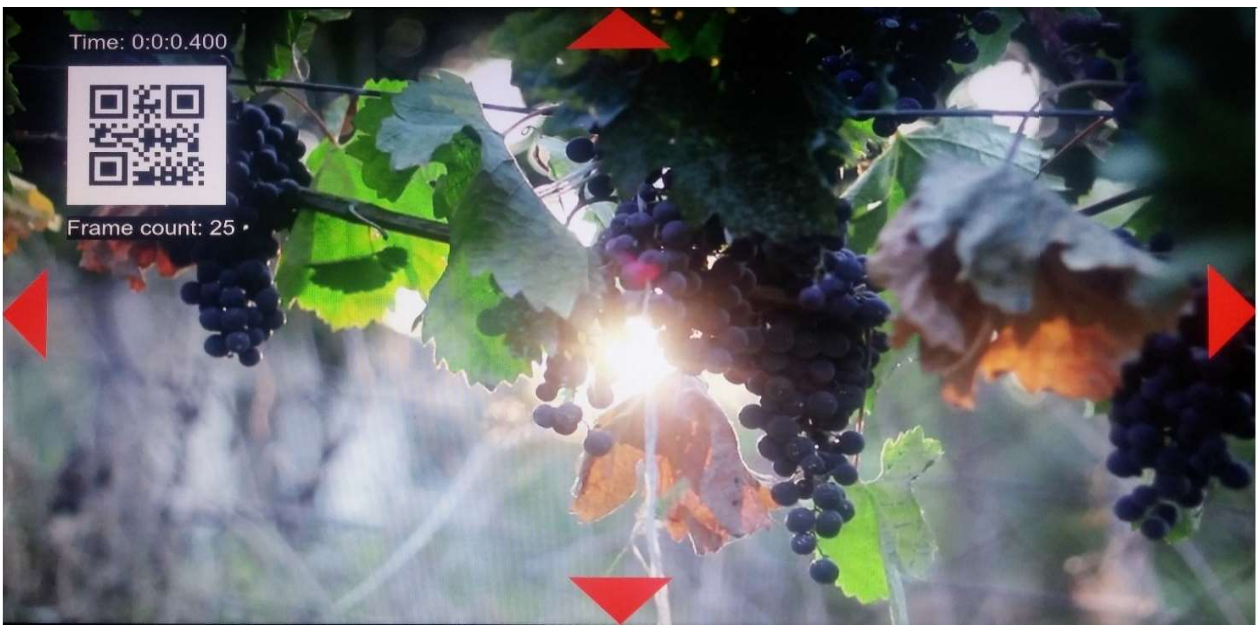
Zbog toga treba postaviti neka ograničenja pri snimanju kamerom kako bi se smanjila pojava elemenata koji nisu potrebni za rada algoritma i analizu video okvira te kako bi se održala preciznost položaja elemenata video okvira. Ako nema ograničenja u načinu snimanja može se dogoditi da se podnaslov pojavi na koordinatama koje u velikoj mjeri odstupaju od pravih koordinata što je uzrokovano prevelikim kutom snimanja kamere ili prevelikom udaljenošću od zaslona.

Ovaj zahtjev je riješen dodavanjem strelica koje pokazuje rubove video okvira (Sl. 3.6). Te strelice služe kao vodilje kada se snimanje zaslona vrši koristeći kameru.



Slika 3.6 Strelice vodilje za snimanje kamerom

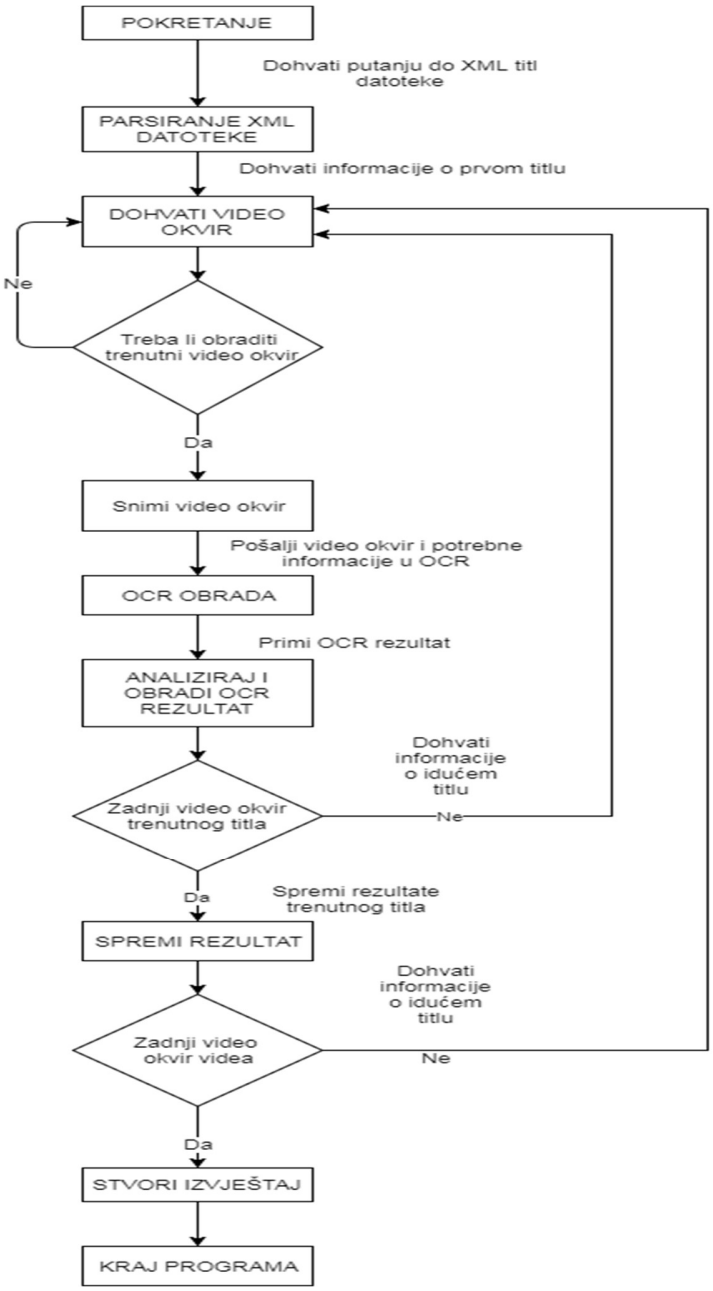
Kameru je potrebno pozicionirati tako da strelice budu na krajevima prostora koji je obuhvaćen kamerom. U slučaju da je nemoguće pozicionirati kameru da su sve strelice postavljene na rubove, minimalni uvjet je da se kamera postavi tako da su lijeva strelica i donja strelica pozicionirane na rubove. Ako se ni taj uvjet ne može zadovoljiti nemoguće je garantirati preciznost rada algoritma. Primjer pravilno pozicionirane kamere u odnosu na ekran je dan na slici 3.7.



Slika 3.7 Primjer pravilno pozicionirane kamere

4. NAČIN RADA ALGORITMA ZA PROVJERU SINKRONIZACIJE VIDEO SADRŽAJA I SUBTITLOVA

Algoritam je podijeljen na tri glavne funkcije. Te funkcije obavljaju čitanje XML datoteke, analizu video okvira te generiranje izvještaja. Dijagram toka algoritma je dan na slici 4.1, a pojedinačni opis modula u daljnjem tekstu.



Slika 4.1 Dijagram toka algoritma za provjeru sinkronizacije video sadržaja i podnaslova

4.1 Modul za čitanje XML datoteke

Funkcija za čitanje XML datoteke je prvi korak u radu algoritma. Funkcija prima putanju do XML podnaslov datoteke te ju analizira na osnovu definiranih XML oznaka koji su zapisani u njoj i te podatke sprema u za to napravljene strukture. Napravljene su dvije podatkovne strukture od kojih jedna u sebe sprema generalne podatke o XML podnaslov datoteci kao što su jezik i boja fonta, a u drugu strukturu se unutar niza struktura spremaju podaci o svakom pojedinačnom podnaslovu unutar XML podnaslov datoteke. Ti podaci su početni i krajnji video okvir podnaslova kao i njegov položaj te tekst. Tako obrađena XML datoteka je spremna za rad s glavnim dijelom algoritma za obradu video okvira.

4.2 Modul za analizu video okvira

Modul za analizu video okvira prima putanju do video datoteke i broj koji predstavljanje odstupanje unutar kojeg će se analizirati video okviri. Na primjer, ako podnaslov ima početak na video okviru broj 6 i kraj na video okviru broj 22 a vrijednost odstupanje je zadana u vrijednosti od 3 video okvira, algoritam će analizirati video okvire u rasponu od video okvira broj 3 do video okvira broj 9 za početak podnaslova te video okvire u rasponu od video okvira broj 19 do video okvira broj 25 za kraj podnaslova. Ovo se koristi kako bi utvrdili da li je podnaslov počeo prije zadanog vremena početka ili završio poslije zadanog vremena završetka. Nakon primanja potrebnih informacija modul počinje čitati svaki video okvir te analizira vrijednost koja je zapisana u QR kodu kako bi znao na kojem video okviru se nalazi. Ukoliko detektira da se trenutni video okvir nalazi unutar raspona početka podnaslova uvećanog to jest umanjenog za broj odstupanja video okvir ili kraja podnaslova uvećanog to jest umanjenog za broj odstupanja video okvir snima trenutni video okvir te ga predaje programu za optičko prepoznavanje znakova (*eng. Optical Character Recognition* - OCR) na analizu.

Nakon što OCR vrati rezultate poziva se funkcija koja analizira rezultate te na osnovi analize postavljaju vrijednosti detektiranog početka i kraja podnaslova. Algoritam uvijek sprema broj video okvira u kojem se prvi puta pojavio podnaslov te broj video okvira na kojem se zadnji puta pojavio podnaslov. Ukoliko detektira zadnji video okvir u rasponu koji je potrebno analizirati poziva se funkcija koja sprema rezultat u strukturu koja sadržava rezultate analiziranih podnaslova. Na osnovu podataka zapisanih u toj strukturi se poziva funkcija koja vrši provjeru jesu li podnaslovi sinkronizirani, te ukoliko nisu sinkronizirani, određuje se na koji način nisu u sinkronizaciji.

Točnije, određuje se da li detektirani početni i krajnji video okviri na kojim se pojavljuje podnaslov odgovaraju vremenima definiranim u XML podnaslov datoteci. Ukoliko ne odgovaraju, provjerava se je li razlog u preranom, to jest prekasnom početku ili kraju pojavljivanja podnaslova. Nakon spremanja rezultata, prebacuje se početno i završno vrijeme na vrijeme sljedećeg podnaslova za koji treba provjeriti sinkronizaciju te se ponavlja postupak za njega. Nakon što se analiziraju svi video okviri video datoteke, poziva se modul za generiranje izvještaja i zatvaranje video datoteke.

4.3 Modul za generiranje izvještaja

Modul za generiranje izvještaja prima podatke o svakom početnom i krajnjem detektiranom pojavljivanju podnaslova za svaki obrađeni video okvir od modula za analizu video okvira. Nakon primanja podataka analiziraju se dobivene vrijednosti i na osnovu njih se postavljaju vrijednosti unutar strukture koja sadržava rezultate za svaki podnaslov. Na osnovu definiranih kodova provjerava da li se pojavila greška u čitanju QR koda te da li je detektiran početak i kraj podnaslova ili se podnaslovi uopće nisu pojavili na video okviru. Nakon analize svih potrebnih video okvira i podnaslova, generira se izvještaj u obliku .csv datoteke (*eng. Comma Separated Values* - CSV). Unutar izvještaja je zapisan podatak o početnom i krajnjem video okviru na kojem se pojavljuje podnaslov, te informacije da li je podnaslov sinkroniziran sa svojim pratećom XML podnaslov datotekom.

5. REZULTATI TESTIRANJA

5.1 Skripta za generiranje testnih sekvenci

Kako bi se moglo početi s testiranjem zadatka prvo je potrebno izraditi testne sekvence na kojima će se testirati algoritam. Za potrebe testiranja izrađena je skripta za generiranje testnih sekvenci na Node.js platformi. Skripta prilikom generiranja testnih video sekvenci također generira i prateći XML podnaslov datoteku u kojoj se automatski upisuju potrebni podaci koje će algoritam analizirati.

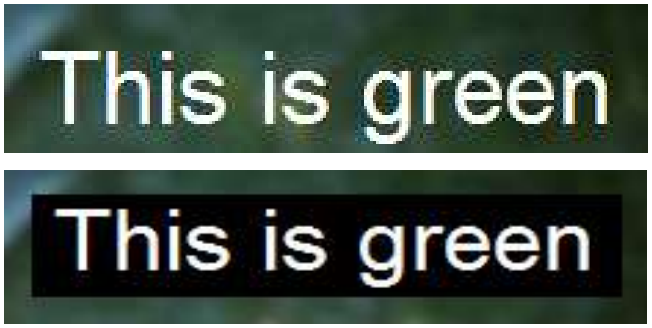
Prvi dio skripte se bavi izradom video datoteke s potrebnim informacijama. Prvi korak pri izradi testnih sekvenci je rastavljanje željene video datoteke na video okrive (Sl. 5.1). Taj postupak rastavljanja se obavlja pomoću *FFmpeg* okružja i pokreće se iz komandne linije te kao ulazne parametre prima naziv video datoteka nad kojom će se vršiti rastavljanje, broj okvira po sekundi video datoteke koju se rastavlja te putanju do mjesta snimanja rastavljenih video okvira.

Slika 5.1 Jedan video okvir nakon rastavljanja video datoteke

Nakon ekstrakcije video okvira potrebno ih je obraditi i dodati potrebne informacije na njih. Pokretanjem skripte za generiranje pokreče se automatizirani postupak koji mijenja rastavljene video okvire i na njih lijepi informacije o broju video okvira u obliku QR koda, postavlja sa svake strane strelice koje služe kao vodilje za snimanje zaslona kamerom, te lijepi tekst podnaslova na video okvir pri čemu je položaj te trenutak pojavljivanja i završetka generiran unutar skripte.

Skripta za generiranje testnih sekvenci se pokreče iz komandne linije pri čemu kao jedini ulazni podatak prima broj video okvira koje treba generirati. Moguće je unijeti manji broj video okvira nego što je broj video okvira video datoteke koje smo rastavili, što će rezultirati kraćom video datotekom nakon što se video okviri ponovo spoje u video datoteku. Također je i obrnuti postupak moguć, ali će svi video okviri koji su izvan broj video okvira izvorne video datoteke biti generirani kao video okviri s potpuno crnom pozadinom.

Mogu se generirati video okviri gdje tekst podnaslova ima pozadinu ili nema pozadinu (Sl. 5.2).



Slika 5.2 Primjer podnaslova sa i bez pozadine

Nakon što skripta grafički obradi video okvire, video okviri poprimaju oblik prikazan na slici 5.3. te su spremni za sastavljanje u video datoteku koja će se koristiti za testiranje algoritma.

Slika 5.3 Jedan video okvir nakon obrade skriptom za generiranje sekvenci

Sastavljanje video okvira u video datoteku koja će se koristiti kao testna sekvenca se također vrši pomoću *FFmpeg*-a. Naredba koja obavlja sastavljanje video okvira u novu video datoteku kao parametre komandne linije prima okvirnu stopu (*eng. frame rate*) koju će imati generiran video datoteka, putanju do obrađenih video okvira, bitnu brzinu (*eng. bit rate*) te naziv koji će imati generirana video datoteka. Nakon pokretanja te naredbe dobije se video datoteka koje je spremna za testiranje razvijenim algoritmom.

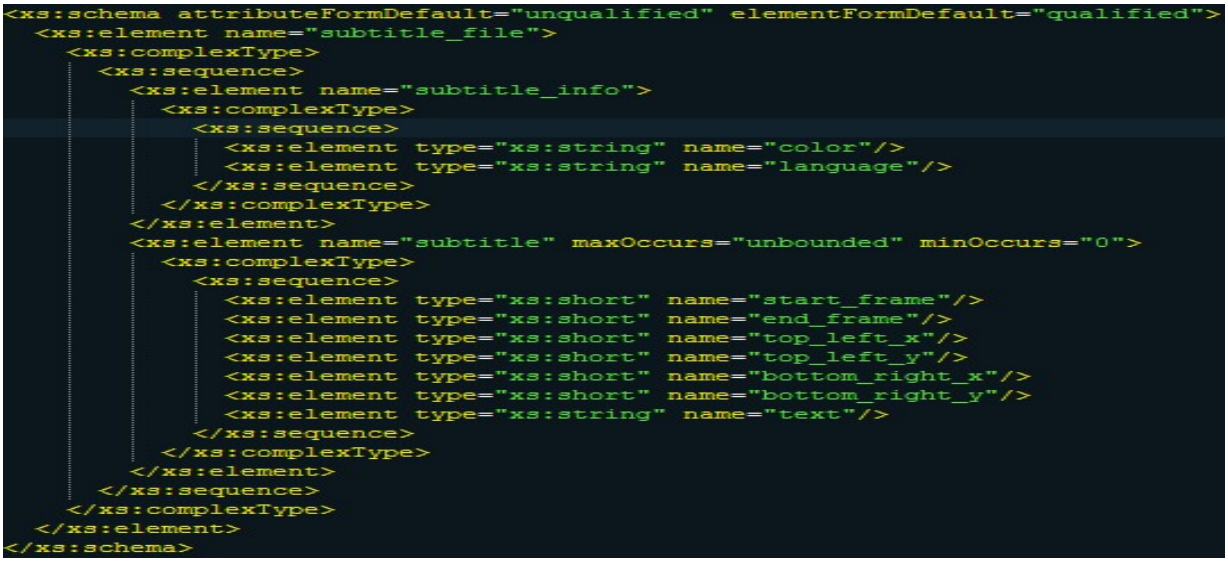
Drugi dio skripte se bavi izradom prateće XML podnaslov datoteke. XML podnaslov datoteka u sebi sadrži potrebne informacije do kojih algoritam dolazi čitanjem (*eng. parsing*) XML datoteke.

Unutar XML datoteke su zapisane informacije o:

- Boji fonta
- Jeziku prijevoda
- Početku i kraju titla
- Položaj titla

- Tekst titla

Interna struktura XML datoteke je prikazana na slici 5.4.



Slika 5.4 Struktura XML titl datoteke

Podaci koji su upisani u XML datoteci se automatski generiranju pomoću algoritma unutar skripte koji određuje na kojem video okviru će se pojaviti podnaslov te na kojem video okviru će završiti podnaslov nakon toga određuje položaj na kojem će se pojaviti taj podnaslov te na kraju generira tekst koji će se prikazati. Nakon što generira svaki pojedini dio te podatke zapisuje u XML datoteku koristeći NPM paket za Node.js koji se zove XML „*xml-builder*“. On omogućuje jednostavno kreiranje i manipulaciju XML datoteke i njegovih elemenata. Takva gotova XML datoteka je spremna za rad s razvijenim algoritmom.

5.2 Testiranje na video sekvencama

Testiranje algoritma se vrši nad video datotekama koje imaju svoje prateće XML podnaslov datoteke. Pokretanjem algoritma za provjeru sinkronizacije videa sadržaja i podnaslova se dobije izvještaj u obliku .svg datoteke iz koje se jasno vidi da li je test prošao, to jest da li je algoritam točno detektirao sva vremena pojavljivanja i kraja podnaslova koji su definirani u XML podnaslov datoteci. Ukupno će se izvršiti deset testova podijeljenih u dvije kategorije, kategorija bez OCR grešaka i kategorija s OCR greškama. U prvoj kategoriji se vrši sedam testova uz pretpostavku da korišteni OCR program radi sa 100 % točnošću to jest bez grešaka kako bi se fokus testiranja stavio

na vremena početka i kraja pojavljivanja podnaslova. U drugoj kategoriji se vrše tri testa s pretpostavkom da OCR program ne radi sa 100 % točnošću to jest s greškama na način da tekst koji on pročita ne odgovara traženom tekstu. Svaka od karakterističnosti pojedinog testa je opisana prije svakog pojedinog testa.

Vremena pojavljivanja podnaslova na video okvirima je potrebno modificirati kako bi se stvorile namjerne pogreške u odnosu na vrijeme pojavu podnaslova unutar prateće XML podnaslov datoteke te je potrebno provjeravati da li algoritam pravilno detektira vremena pojave podnaslova na razini video okvira. Svako testiranje je izvršavano s vrijednošću odstupanja od 5 video okvira kako bi uvjeti provođenja svakog testa bilo identični. Broj video okvira za odstupanje je uzet proizvoljno te je mogao biti bilo koja druga vrijednost.

Kod testova u prvoj kategoriji svaka osim prve sekvence se testira dva puta, jednom s video datotekom koja potječe s uređaja za hvatanje sadržaja u 30 okvira po sekundi te video datotekom iste te sekvence snimane kamerom u 60 okvira po sekundi. Prva sekvenca nije testirana dva puta jer je ona snimana uređajem za hvatanje sadržaja u 60 okvira po sekundi te je nema potreba ponovo snimati kamerom u 60 okvira po sekundi. Testovi 2 i 3, 4 i 5 te 6 i 7 se trebaju promatrati kao cjelina jer bi rezultati testova trebali biti jednaki ako algoritam pravilno detektira sve početke i krajeve pojavljivanja podnaslova jer su testovi identični samo se način snimanja video datoteke razlikuje. Razlog ovakvog testiranja je taj kako bi se uz preciznost rada algoritma utvrdio i utjecaj kvalitete slike snimane kamerom na rad algoritma. U drugoj kategoriji se svaki test izvršava samo jednom.

5.2.1 Testiranja bez OCR grešaka

Karakterističnosti svih testova unutar ove kategorije su dana u tablici 5.1. U tablici se mogu vidjeti razlike u očekivanim vremenima pojave i kraja podnaslova u odnosu na stvarna vremena pojavljivanja za svaki pojedini test unutar ove kategorije. Detaljnije o broju odstupanja video okvira unutar svakog testa je dano u tablicama postavki pojedinog testa.

Test broj	**Karakterističnost testa**	
	Podnaslovi u sinkronizaciji	*Podnaslovi izvan sinkronizacije*
1	3,7,8	1,2,4,5,6,9,10,11,12
2,3	1,2,3,4	-
4,5	-	1,2,3
6,7	2	1,3

Tablica 5.1 Karakterističnosti pojedinih testova u prvoj kategoriji

Test 1 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 60 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 12 podnaslova čija su vremena pojavljivanja i završetka definirana u tablici 5.2.

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	28	300	28	303
2	392	508	390	508
3	602	791	602	791
4	861	989	859	992
5	1107	1358	1107	1356
6	1473	1617	1477	1620
7	1760	1987	1760	1987
8	2132	2349	2132	2349
9	2469	2670	2466	2670
10	2793	2966	2795	2964
11	3027	3194	3025	3197
12	3277	3331	3275	3328

Tablica 5.2 Postavke za Test 1

Rezultat testiranja:

Rezultati Testa 1 su dani u tablici 5.3. Algoritam je detektirao vremena početka i kraja svakog podnaslova sa 100 % točnošću te odredio točan tip sinkronizacije za svaki podnaslov.

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
28	303	+/-
390	508	-/+
602	791	+/+
859	992	-/-
1107	1356	+/-

1477	1620	-/-
1760	1987	+/+
2132	2349	+/+
2466	2670	-/+
2795	2964	-/-
3025	3197	-/-
3275	3328	-/-

Tablica 5.3 Rezultati za Test 1

Test 2 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 30 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 4 podnaslova. Vremena početka i kraja svih podnaslova su postavljena da odgovaraju vremenima u XML podnaslov datoteci (Tab. 5.4).

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	24	207	24	207
2	342	496	342	496
3	623	750	623	750
4	885	922	885	922

Tablica 5.4 Postavke za Test 2

Rezultat testiranja:

Rezultati Testa 2 su dani u tablici 5.5. Algoritam je detektirao vremena početka i kraja svakog podnaslova sa 100 % točnošću te odredio točan tip sinkronizacije za svaki podnaslov.

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
24	207	+/+
342	496	+/+
623	750	+/+
885	922	+/+

Tablica 5.5 Rezultati za Test 2

Test 3 se izvršava nad video datotekom snimljenom kamerom u 60 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 4 podnaslova. Vremena početka i kraja svih podnaslova su postavljena da odgovaraju vremenima u XML podnaslov datoteci (Tab. 5.6).

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	24	207	24	207
2	342	496	342	496
3	623	750	623	750
4	885	922	885	922

Tablica 5.6 Postavke za Test 3

Rezultat testiranja:

Rezultati Testa 3 su dani u tablici 5.7. Algoritam je detektirao vremena početka i kraja svakog podnaslova sa 100 % točnošću te odredio točan tip sinkronizacije za svaki podnaslov.

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
24	207	+/+
342	496	+/+
623	750	+/+
885	922	+/+

Tablica 5.7 Rezultati za Test 3

Test 4 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 30 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 3 podnaslova. Vremena početka i kraja svih podnaslova su postavljena tako da je odmak toliko velik da svi podnaslovi počnu i završe prije vremena početka definiranih unutar XML podnaslov datoteke (Tab. 5.8). Tako je testirano da li algoritam pravilno detektira da se podnaslovi nisu pojavili, a time je indirektno testirano da li algoritam detektirati i ako se podnaslovi uopće ne pojave na video okvirima.

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	27	312	15	21
2	406	683	350	399
3	823	966	760	800

Tablica 5.8 Postavke za Test 4

Rezultat testiranja:

Algoritam je sa 100 % točnošću detektirao da se podnaslovi nisu pojavili unutar predviđenog vremena definiranog u XML podnaslov datoteci (Tab. 5.9).

Sub start frame	Sub end frame	Detected start frame	Detected end frame	Sub sync type
27	312	Not detected	Not detected	-/-
406	683	Not detected	Not detected	-/-
823	966	Not detected	Not detected	-/-

Test 5 se izvršava nad video datotekom snimanom kamerom u 60 okvira po sekundi. Postavke testa su identične postavkama testa 4 i prikazane su u tablici 5.10.

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	27	312	15	21
2	406	683	350	399
3	823	966	760	800

Tablica 5.9 Postavke za Test 5

Rezultat testiranja:

Algoritam je sa 100 % točnošću detektirao da se podnaslovi nisu pojavili unutar predviđenog vremena definiranog u XML podnaslov datoteci (Tab. 5.11).

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
Not detected	Not detected	-/-
Not detected	Not detected	-/-
Not detected	Not detected	-/-

Tablica 5.10 Rezultati za Test 5

Test 6 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 30 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 3 podnaslova. Vremena početka i kraja podnaslova 2 su postavljeni da odgovaraju vremenima u XML datoteci, dok su u podnaslovima 1 i 3 napravljena odstupanja od definiranih vremena u XML datoteci (Tab. 5.12).

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	23	268	25	270
2	359	511	359	511
3	578	754	576	751

Tablica 5.11 Postavke za Test 6

Rezultat testiranja:

Algoritam je sa 100 % točnošću detektirao da je podnaslov 2 počeo i završio u pravo vrijeme te pravilno detektirao odstupanja nastala u podnaslovima 1 i 3 (Tab. 5.13).

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
25	270	-/-
359	511	+/+
576	751	-/-

Tablica 5.12 Rezultati za Test 6

Test 7 se izvršava nad video datotekom snimanom kamerom u 60 okvira po sekundi. Postavke testa su identične postavkama testa 6 i prikazane su u tablici 5.14.

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Početak na video okviru	Kraj na video okviru
1	23	268	25	270
2	359	511	359	511
3	578	754	576	751

Tablica 5.13 Postavke za Test 7

Rezultat testiranja:

Algoritam je sa 100 % točnošću detektirao sva vremena početka i kraja podnaslova (Tab. 5.15).

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
25	270	-/-
359	511	+/+
576	751	-/-

Tablica 5.14 Rezultati za Test 7

5.2.2 Testiranje s OCR greškama

Karakterističnosti svih testova unutar ove kategorije su dana u tablici 5.16. Za razliku od testova koji su se vršili u prvoj kategoriji sva vremena početka i kraja podnaslova su ostavljena u sinkronizaciji sa svojom XML datotekom, ali određeni tekstovi podnaslova prikazanim na video okvirima su neispravni. Na taj način se testira da li algoritam pravilno detektira da se traženi tekst nije pojavio na video okviru ili pročitani tekst ne odgovara traženom tekstu.

Test broj	Karakterističnost testa	
	Podnaslovi s ispravnim tekstom	*Podnaslovi s pogrešnim tekstom*
8	1,4	2,3
9	-	1,2,3
10	1,2,3	-

Tablica 5.15 Karakterističnosti pojedinih testova u drugoj kategoriji

Test 8 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 30 okvira po sekundi. Tekstovi podnaslova 2 i 3 na video okviru ne odgovaraju tekstovima u XML podnaslov datoteci dok tekstovi podnaslova 1 i 4 odgovaraju. Postavke testa su prikazane u tablici 5.17.

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Tekst ispravan
1	24	207	Da
2	342	496	Ne
3	623	750	Ne
4	885	922	Da

Tablica 5.16 Postavke za Testa 8

Rezultat testiranja:

Algoritam je sa točno detektirao da se tekst podnaslova 1 i 2 odgovaraju tekstu u XML podnaslov datoteci a da tekst podnaslova 2 i 3 ne odgovara (Tab. 5.18).

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
24	207	+/+
Not detected	Not detected	-/-
Not detected	Not detected	-/-
885	922	+/+

Tablica 5.17 Rezultati za Test 8

Test 9 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 30 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 3 podnaslova. Svaki podnaslov ima grešku u tekstu pri prikazu na video okviru (Tab. 5.19).

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Ispravan tekst
1	27	312	Ne
2	406	683	Ne
3	823	966	Ne

Tablica 5.18 Postavke za Test 9

Rezultat testiranja:

Algoritam je točno detektirao da svi pročitani tekstovi ne odgovaraju očekivanim tekstovima unutar XML podnaslov datoteke (Tab. 5.20).

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
Not detected	Not detected	-/-
Not detected	Not detected	-/-
Not detected	Not detected	-/-

Tablica 5.19 Rezultati za Test 9

Test 10 se izvršava nad video datotekom snimljenom uređajem za hvatanje sadržaja u 30 okvira po sekundi. Prateća XML podnaslov datoteka se sastoji od 3 podnaslova. Svaki tekst podnaslova na video okviru odgovara očekivanim tekstovima unutar XML podnaslov datoteke. Postavke testa su prikazane u tablici 5.21.

Redni broj podnaslova	Početak definiran u podnaslov datoteci	Kraj definiran u podnaslov datoteci	Ispravan tekst
1	23	268	Da
2	359	511	Da
3	578	754	Da

Tablica 5.20 Postavke za Test 10

Rezultat testiranja:

Algoritam je točno detektirao da su svi tekstovi točni i vremena pojavljivanja točna. (Tab. 5.22).

Detektirani početak na video okviru	Detektirani početak na video okviru	Tip sinkronizacije
23	268	+/+
359	511	+/+
578	754	+/+

Tablica 5.21 Rezultati za Test 10

6. ZAKLJUČAK

Sinkronizacija video sadržaja i podnaslova je jedan od najbitnijih dijelova dostavljanja nekog video sadržaja krajnjim korisnicima. Ako ta dva elementa nisu sinkronizirani kako treba, stvara se nelagoda prilikom gledanja video sadržaja kod krajnjih korisnika. Iz tog razloga se i izrađivao prikazani algoritam, u svrhu osiguranja kvalitete sadržaja.

Zadatak diplomskog rada je bio realiziranje algoritma za provjeru sinkronizacije video sadržaja i podnaslova. Aplikacija je izrađena u programskom jeziku C te je koncipirana kao 3 odvojena modula od kojih se svaki bavi pojedinim područjem analize i izrade izvještaja.

Aplikacije je testirana na sedam odvojenih testova koji su testirali preciznost rada algoritma u raznim kombinacijama stanja prikaza podnaslova. Svaki od provedenih testova u obje kategorije je uspješno izvršen. Potrebno je naglasiti da su se sva testiranja u prvoj kategoriji provodila pod pretpostavkom da OCR program radi sa 100 % točnošću.

Iako aplikacija u potpunosti izvršava svoju namjenu, ona je namijenjena da radi u strogo kontroliranim uvjetima, tako da ima mjesta za napredak algoritma kako bi se još povećala preciznost i proširila mogućnost uporabe algoritma u što više uvjeta i situacija. Također kvaliteta kamera i broj okvira po sekundi koje kamera snima ima velik utjecaj na preciznost algoritma jer se povećanjem ponovljenih video okvira povećava i broj video okvira sa kojih se može pravilno dekodirati QR kod s upisanim brojem trenutnog video okvira.

LITERATURA

[1] „Overview | HbbTV“. [Na internetu]. Dostupno na: https://www.hbbtv.org/overview/. [Pristupljeno: 30-kol-2017].

[2] Brian W. Kernighan i D. M. Ritchie, *The C Programming Language*, Second Edition. New Jersey: PRENTICE HALL.

[2] Node.js Foundation, „Node.js“, *Node.js*. [Na internetu]. Dostupno na: https://nodejs.org/en/. [Pristupljeno: 30-kol-2017].

[4] „Extensible Markup Language (XML)“. [Na internetu]. Dostupno na: https://www.w3.org/XML/. [Pristupljeno: 30-kol-2017].

[5] „ffmpeg Documentation“. [Na internetu]. Dostupno na: https://ffmpeg.org/ffmpeg.html. [Pristupljeno: 30-kol-2017].

[6] „QR Code Overview and Progress of QR Code Applications“. GS1 Japan, 2009.

[7] „QR Code Essentials“. DENSO Wave Incoperated, 2012.

[8] C. C.K.P, „Reed-Solomon error correction“. British Brodcasting Corporation R&D, srp-2012.

SAŽETAK

U ovom radu je izrađen algoritam za provjeru sinkronizacije video sadržaja i podnaslova. Algoritam je pisan u programskom jeziku C. U radu su opisane sve korištene tehnologije koje su bile potrebne za realizaciju algoritma. Te tehnologije su programski jezik C, XML označni jezik, Node.js okružje, *FFmpeg* te tehnologija dvodimenzionalnih QR kodova. Dan je opis rješenja zadataka, opis pojedinih modula, njihova međuovisnost te provedeni testovi i njihovi rezultati. Algoritam je izrađen u tri modula, a ti moduli su modul za iščitavanje XML datoteke, modula za analizu video okvira te modul za generiranje izvještaja. Testiranje je izvršeno na deset različitih testnih sekvenci i pomoću njih je testirana preciznost algoritma da detektira sinkronizaciju video sadržaja i podnaslova. Testiranjem je zaključeno da algoritam radi ispravno te da je zadani zadatak uspješno obavljen.

Ključne riječi: C, XML, FFmpeg, QR kod, Node.js, sinkronizacija, podnaslov, video okvir, kamera, okvir po sekundi, početak, kraj

ABSTRACT

In this paper the algorithm for checking the video content and subtitle synchronization was created. The algorithm is written in the programming language C. The paper describes all technologies used for algorithm realization. These technologies are program language C, XML markup language, Node.js, *FFmpeg*, and two-dimensional QR code technology. The description of the solution, individual modules, their interdependence and tests are performed and their results are presented in the paper. The algorithm was created in the three modules, and these modules are modules for reading XML files, module for the video frame analysis, and report generation module. The testing was performed on ten different test sequences. By running these tests, the algorithm precision of video content detecting and subtitle synchronization was tested. After the testing is performed, it is concluded that the algorithm is working correctly and that the given task has been successfully completed.

Keywords: C, XML, FFmpeg, QR Code, Node.js, synchronization, subtitle, video frame, camera, frames per second, beginning, end

ŽIVOTOPIS

Davor Kedačić je rođen 22.9.1989. godine u Đakovu. Do svoje 9. godine živi u Budrovcima, obližnje selo Đakova, gdje završava prvi i drugi razred osnovne škole. 1998. godine seli u mjesto Batinu, na samoj granici Hrvatske i ondje 2004. godine završava osnovnu školu i upisuje srednju školu u Belom Manastiru u smjeru – Tehničar za računalstvo. 2008. godine završava srednju školu i upisuje Elektrotehnički fakultet u Osijeku, preddiplomski sveučilišni studij, smjer Računarstvo. 2015. godine završava preddiplomski studij te iste godine upisuje Diplomski sveučilišni studij, smjer Računalno inženjerstvo.

_____

Davor Kedačić

PRILOZI