Zbornik šeste Elektrotehniške in računalniške konference ERK'97

25.-27. september 1997
Portorož, Slovenija

Zvezek B

Računalništvo in informatika
Umetna inteligenca
Robotika
Razpoznavanje vzorcev - 3. letna konferenca SDRV
Biomedicinska tehnika
Močnostna elektrotehnika
Didaktika
Študentski članki

Uredil
Baldomir Zajc

IEEE Region 8
Slovenska sekcija IEEE
Ljubljana • Slovenija

Testiranje algoritmov s področja računalniškega vida preko svetovnega spleta *

Danijel Skočaj, Aleš Jaklič, Aleš Leonardis, Franc Solina
Laboratorij za računalniški vid, Fakulteta za računalništvo in informatiko
Univerza v Ljubljani
Tržaška 25, 1001 Ljubljana, Slovenija
danijels@razor.fer.uni-lj.si

Povzetek

V tem delu smo raziskali različne možnosti uporabe interneta, s katerimi uporabnikom omogočimo dostop do algoritmov, ki jih razvijamo. Opisali smo postopek za izgradnjo interaktivne aplikacije, ki deluje po modelu odjemalec/strežnik in za komunikacijo uporablja svetovni splet. Odjemalni program je programček v Javi, strežni program pa deluje na strežniku kot CGI program, ki ga na zahtevo odjemalca zažene HTTP strežnik. Tudi prenos podatkov med odjemalnim in strežnim programom poteka preko HTTP strežnika, saj se za prenos uporablja HTTP protokol. Samostojno delujoči program za segmentacijo slik smo priredili v aplikacijo, ki deluje po modelu Java-odjemalec/CGI-strežnik in je uporabnikom na voljo kot storitev na svetovnem spletu.

1 Uvod

Raziskovalci na področju računalniškega vida uporabljajo različne računalnike in operacijske sisteme. Zato je težko omogočiti uporabo programske opreme na platformah, ki se razlikujejo od tiste, na kateri je bila programska oprema razvita.

Na srečo imamo internet, ki omogoča komunikacijo med različnimi platformami. Raziskovalci si lahko izmenjujejo sporočila, razpravljajo o problemih v diskusijskih skupinah in izmenjujejo celo programe in programsko kodo. Toda še vedno ostaja problem - kako omogočiti izvajanje teh programov na različnih platformah. Prevajanje izvorne kode je le delna rešitev. Za učinkovito rešitev tega problema potrebujemo platformno neodvisni jezik.

Svetovni splet ponuja rešitev - programski jezik Java [1]. Koda, ki jo dobimo s prevajanjem programa napisanega v Javi, je popolnoma platformno neodvisna in prenosljiva ter jo lahko interpretirajo spletni pregledovalniki na vseh pomembnejših operacijskih sistemih. V Javi lahko napišemo kar algoritme, lahko pa v Javi napišemo samo vmesnik do programa, ki se izvaja na našem računalniku. Na ta način omogočimo oddaljeno testiranje naših algoritmov z različnih platform.

V tem delu smo raziskali različne možnosti uporabe interneta, s katerimi lahko algoritme naredimo javno dosegljive. Opisali smo postopek za izgradnjo interaktivne aplikacije, ki deluje po modelu odjemalec/strežnik, kjer je odjemalec programček v Javi (Java Applet), strežnik pa CGI (Common Gateway Interface [2]) program. To tehnologijo smo uporabili na primeru programa za segmentacijo slik, ki je sedaj na voljo kot servis na svetovnem spletu.

2 Programska oprema s področja računalniškega vida na internetu

Dostop do naših algoritmov preko interneta lahko omogočimo na različne načine [3].

Distribucija binarne kode

Distribucija binarne kode z objavo na anonimnem FTP strežniku je zelo pogost način uporabe interneta za omogočanje javnega dostopa do programov. Vsakdo lahko naloži program na svoj računalnik, toda izvajajo ga lahko le uporabniki, ki delajo na platformi za katero je bil program napisan oz. preveden. Program je popolnoma platformno odvisen. Lahko se sicer potrudimo in program prevedemo na več različnih platformah, vendar to prav gotovo ni univerzalna rešitev.

Distribucija izvorne kode

Anonimni FTP strežnik lahko uporabimo tudi za distribucijo izvorne kode. Vsakdo lahko to kodo naloži, jo prevede, požene in testira. Toda med prevajanjem izvorne kode se pogosto pojavljajo težave z različnimi nastavitvami, knjižnicami in platformno specifičnimi funkcijami. S previdnim programiranjem

* To delo so podprli Ministrstvo za znanost in tehnologijo Republike Slovenije (Projekti J2-6187, J2-8829, L2-8721), Copernicus Program Evropske unije (Grant 1068 RECCAD) in Ameriško – slovenski projekt (Projekt #95-158).

je mogoče doseči visoko stopnjo prenosljivosti izvorne kode, toda program še vedno ne deluje na vseh platformah, na nekaterih pa deluje samo z dodatnimi napori uporabnikov. Poleg tega na ta način omogočimo uporabnikom ne samo testiranje izvajanja naših algoritmov, temveč tudi vpogled v samo izvorno kodo, kar pa si včasih ne želimo.

Algoritmi v Javi

Če hočemo imeti popolnoma platformno neodvisen program, ga moramo napisati v platformno neodvisnem in prenosljivem jeziku, kot je Java. Uporabnik pride do takega programa na zelo enostaven način. Njegov spletni pregledovalnik naloži skupaj s spletno stranjo tudi programček v Javi in ga zažene. Program se izvaja na uporabnikovem računalniku, ne da bi ga uporabnik posebej namestil.

Toda Java ima tudi nekatere pomanjkljivosti. Ker se programi napisani v Javi interpretirajo, je njihovo izvajanje počasno. Hitrost izvajanja programov pa je na področju računalniškega vida ključna, saj so ponavadi algoritmi računsko zelo zahtevni. Java ima tudi zelo ostre omejitve zaradi varnostnih razlogov. Tako npr. programček (Applet) nima dostopa do datotečnega sistema. Pričakujemo lahko, da se bodo te pomanjkljivosti z razvojem zmanjšale, sedaj pa je programski jezik Java primeren le za implementacijo kratkih, računsko nezahtevnih algoritmov, ki ne potrebujejo dostopa do datotečnega sistema.

Pošiljanje zahtev in čakanje na odgovor

Oddaljeni uporabnik nam lahko preko interneta pošlje samo zahteve in parametre (slike, ukaze). Sami potem te podatke podamo programu, ki jih obdela. Ko je delo končano, obvestimo uporabnika, da lahko naloži rezultate. Ta metoda je zelo nepraktična za uporabo. Celotni proces bi moral biti avtomatiziran.

Pošiljanje zahtev in čakanje na odgovor preko HTML obrazcev

Svetovni splet lahko uporabimo tudi za pošiljanje zahtev in sprejemanje odgovorov. Uporabnik najprej s podatki izpolni HTML obrazec. Ti podatki se potem prenesejo programu na strežniku, ki jih obdela, pretvori rezultate v HTML format in jih pošlje nazaj na uporabnikov računalnik, kjer jih spletni pregledovalnik prikaže.

Ta metoda je primerna za algoritme, ki zahtevajo vnos podatkov samo enkrat, nato podatke obdelajo, ter vrnejo rezultat ravno tako samo enkrat. Primerna je torej samo za algoritme, ki ne zahtevajo višjega nivoja interaktivnosti.

Interaktivne aplikacije po modelu odjemalec/strežnik

V programskem jeziku Java lahko napišemo popolnoma interaktivne aplikacije, ki delujejo po modelu odjemalec/strežnik [4]. Odjemalni program napišemo kot programček v Javi (Java Applet). Ta programček služi kot vmesnik med uporabnikom in računalniškim programom, ki se izvaja kot strežni program na gostiteljskem računalniku. Ta metoda zahteva več truda za implementacijo, toda za rezultat dobimo aplikacijo, ki deluje po modelu odjemalec/strežnik z dvosmerno komunikacijo.

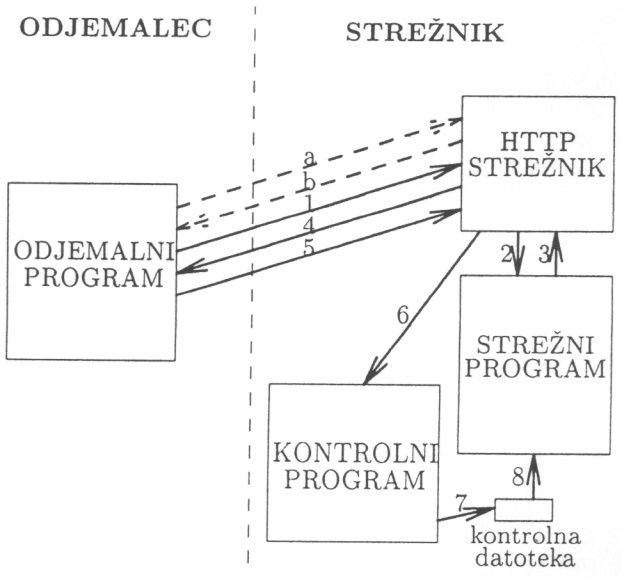
V naslednjem poglavju je opisan način implementacije te metode.

3 Interaktivne aplikacije po modelu odjemalec/strežnik na svetovnem spletu

Značilnost sistema, ki deluje po modelu odjemalec/strežnik je, da del sistema (odjemalec) zahteva izvedbo določenega opravila, drugi del sistema (strežnik) pa to opravilo izvede [5].

Osnovna ideja je, da napišemo odjemalni program kot programček v Javi in osnovni program spremenimo v CGI program. Slednji deluje kot strežni program in izvaja vse računsko zahtevne operacije. Hitrost izvajanja programčka v Javi ni ravno velika, toda ker je odjemalni program kratek in ne vsebuje računsko zahtevnih operacij, je njegovo izvajanje dovolj hitro.

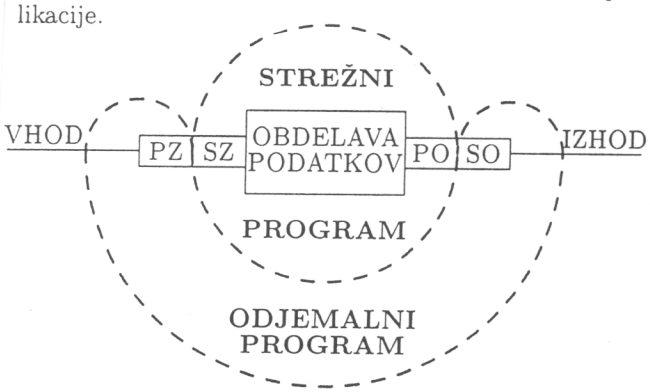
Svetovni splet in njegov HTTP protokol [6] smo uporabili za prenos odjemalnega programa (programčka v Javi) do uporabnika. HTTP strežnik na gostiteljskem računalniku igra vlogo vmesnika med strežnim in odjemalnim programom. Dogajanje je ponazorjeno na Sliki 1.



Slika 1: Komunikacijski model Java-odjemalec/CGI-strežnik.

Uporabnik najprej od našega HTTP strežnika zahteva spletno stran z odjemalnim programom (a) in jo tudi dobi (b). Odjemalni program, ki ga je pognal pregledovalnik, nato zahteva od HTTP strežnika, da sproži CGI program, ki je v bistvu strežni program za storitev (1). Ta to tudi stori (2). Strežni program najprej obvesti HTTP strežnik, da bo odgovore pošiljal v blokih po metodi Strežnik poriva dokument (Server push [7]). HTTP strežnik nato obvesti odjemalca, da je strežni program pripravljen za nadaljnjo komunikacijo (4). Te zveze strežnik ne prekine, saj bo po njej še naprej pošiljal odgovore strežnega programa. To je torej enosmerni komunikacijski kanal od strežnika do odjemalca. Uporabnik pošilja zahteve strežnemu programu na ta način, da preko HTTP strežnika (5) sproži vzporedno še en CGI program in mu hkrati pošlje zahtevo. Ta program zahtevo sprejme (6) in jo prepiše v posebno datoteko (7), od koder jo prebere strežni program. To je torej enosmerni komunikacijski kanal od odjemalca do strežnika. Strežni program te zahteve izpolnjuje in pošilja odgovore preko HTTP strežnika (3) do odjemalca (4). Na ta način je vzpostavljena dvosmerna komunikacija med odjemalcem in strežnikom.

Pri prehodu s samostojno delujočega programa na aplikacijo, ki deluje po modelu odjemalec/strežnik, se interakcija z uporabnikom (vhod) in prikazovanje rezultatov (izhod) preselita na odjemalni program. Na Sliki 2 je shematično prikazan pretok podatkov v takšni aplikaciji. Uporabnik vnaša podatke v odjemalni program. Ta jih potem v skladu s komunikacijskim protokolom pošlje strežnemu programu. Te zahteve so vhod strežnemu programu, ki zahteve obdela ter rezultate pošlje nazaj odjemalcu. Odjemalni program prikaže te rezultate kot končni izhod iz aplikacije.



PZ - Pošiljanje zahtev
SZ - Sprejemanje zahtev
PO - Pošiljanje odgovorov
SO - Sprejemanje odgovorov

Slika 2: Pretok podatkov v programu, ki deluje po modelu odjemalec/strežnik

Ko pretvarjamo osnovni samostojno delujoči program v CGI strežni program, moramo drugače definirati le vhod in izhod, vsi ostali deli programa ostanejo enaki. CGI strežni program bere podatke iz kontrolne datoteke namesto s tipkovnice oz. kakšne druge vhodne enote, zato moramo samostojno delujoči program ustrezno prilagoditi. Podobno velja za izhod. Vse ukaze, ki izpisujejo podatke na zaslon, nadomestimo z ukazi, ki te podatke v ustreznem formatu pošljejo odjemalcu.

Vse te novooblikovane ukaze smo zbrali v razredu *wwwInterface*, ki je programski vmesnik med CGI strežnim programom in HTTP strežnikom. Razred *wwwInterface* poskrbi za komunikacijo v strežnem programu. Ima tudi ustreznega partnerja v odjemalnem programu, ki razume njegov jezik - razred *WWWinterface* v programskem jeziku Java.

Pisanje aplikacij, ki delujejo po modelu Java-odjemalec/CGI-strežnik je veliko lažje, če uporabljamo par *wwwinterface* vmesnikov. Poskrbita namreč za komunikacijo med obema deloma aplikacije. V samostojno delujoči program moramo le vključiti objekt iz razreda *wwwInterface* ter prenesti vhodne in izhodne funkcije v odjemalni program.

4 On-line Segmentor

On-line Segmentor je aplikacija za segmentacijo slik preko svetovnega spleta, ki deluje po modelu Java-odjemalec/CGI-strežnik. Narejena je iz samostojno delujočega programa *Segmentor*. To je objektno orientirano ogrodje za segmentacijo slik, ki temelji na paradigmi rasti in selekcije [8]. Implementirana je segmentacija globinskih slik s superelipsoidi.

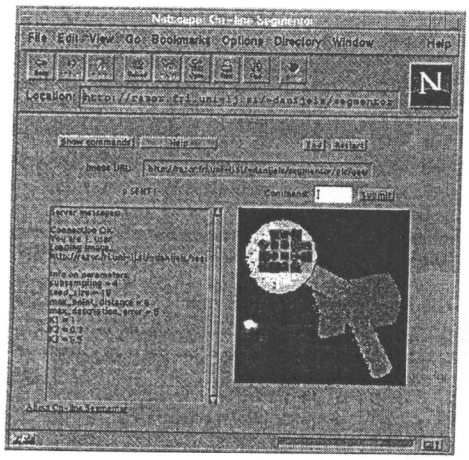
Razred *wwwInterface* smo nadgradili v razred *wwwSegmentor*, ki je podedoval vse lastnosti razreda *wwwInterface* in ima dodatne metode, ki so specifične za program *Segmentor*.

V odjemalnem programu smo uporabili razred *WWWinterface* v Javi. Ta vzpostavi zvezo in skrbi za prenos podatkov med odjemalcem in strežnikom. Da bi dosegli boljši odzivni čas, se namesto primitivnih grafičnih elementov (točk, črt) prenašajo le parametri superelipsoidov in poligonov, ki jih generira strežni program. Odjemalni program zna iz teh parametrov ponovno zgraditi superelipsoide in poligone. V odjemalnem programu sta realizirana tudi vnos podatkov in prikazovanje rezultatov.

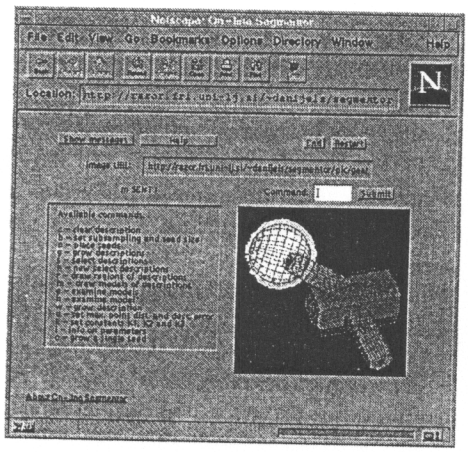
Uporabnik, ki hoče uporabljati program *On-line Segmetor*, mora najprej na svoj računalnik naložiti spletno stran z URL naslova *http://razor.fri.uni-lj.si/danijels/segmentor/segmentor.html*. Ta stran vsebuje odjemalni program, ki ga zažene spletni pregledovalnik. Nato uporabnik vnese URL naslov globinske slike, ki jo želi segmentirati. Slika se naloži na strežnik, nato pa prenese tudi na odjemalca, kjer jo odjemalni program prikaže uporabniku. Uporabnik lahko usmerja tok segmentacije z

razpoložljivimi ukazi (postavitev semen, rast deskripcij, selekcija deskripcij, idr.). Strežni program sproti obvešča uporabnika o poteku segmentacije. Superelipsoidi in poligoni se sproti izrisujejo, ravno tako se sproti izpisujejo informacije v tekstovni obliki. Vnos podatkov ter pošiljanje zahtev potekata vzporedno s sprejemanjem odgovorov in prikazovanjem rezultatov. Uporabnik lahko tako med sprejemanjem rezultatov pošilja nove ukaze, ki so obdelani takrat, ko pridejo na vrsto.

Med spreminjanjem samostojno delujočega programa *Segmentor* v aplikacijo, ki deluje po modelu odjemalec/strežnik, smo skušali doseči tri glavne cilje. To so: podobnost z originalnim programom, kratek odzivni čas in majhni popravki osnovnega programa. Vsi trije cilji so bili doseženi. Delo z *On-line Segmentorjem* je zelo podobno delu z originalnim *Segmentorjem*, odzivni čas se ni bistveno povečal, velik del kode, ki nadgrajuje samostojno delujoči program v CGI strežni program, pa je zajet v razredu *wwwSegmentor*, tako da se koda *Segmentorja* ni veliko spremenila.



(a)



(b)

Slika 3: On-line segmentor (a) med postavitvijo semen in (b) po koncu segmentacije.

5 Zaključek

Pogosto je zelo pomembno, da omogočimo uporabo naših algoritmov preko interneta (npr. za primerjalne teste). V tem delu smo raziskali različne načine, na katere lahko uporabnikom omogočimo dostop do naših algoritmov preko interneta. Opisali smo izgradnjo aplikacije, ki deluje po modelu Java-odjemalec/CGI-strežnik. Uporabili smo implementacijo s HTTP protokolom in HTTP strežnikom. Samostojno delujoči program *Segmentor* smo spremenili v aplikacijo, ki deluje po modelu odjemalec/strežnik in je na voljo uporabnikom kot storitev na svetovnem spletu.

Naš glavni zaključek je, da implementacija takšnih aplikacij za testiranje algoritmov s področja računalniškega vida sicer zahteva nekatere spremembe samostojno delujočega programa in tudi izgradnjo odjemalnega programa v Javi, vendar pa omogoča testiranje teh algoritmov z različnih računalnikov in operacijskih sistemov.

Literatura

[1] A. Newman, et al., *Special Edition Using Java*, Que Corporation, 1996.

[2] D. R. T. Robinson, The WWW Common Gateway Interface Version 1.1, University of Cambridge, February 1996.
http://www.ics.uci.edu/pub/ietf/http/related/draft-robinson-www-interface-01.txt

[3] Computer Vision Home Page, Computer Vision Demos.
http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/v-demos.html

[4] D. Skočaj, *Izvajanje oddaljenega programa preko svetovnega spleta*, Diplomska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 1996.

[5] J. Bloomer, *Power programming with RPC*, O'Reilly & Associates, Inc., 1992.

[6] T. Berners - Lee, F. Fielding, U. C. Irvine, H. Frystyk, Hypertext Transfer Protocol - HTTP/1.0, Informational, May 1996.

[7] An Exploration of Dynamic Document in Netscape 1.1, Netscape Communications Corporation, 1995.
http://www.netscape.com/assist/net_sites/dynamic_docs.html

[8] A. Jaklič, A. Leonardis, F. Solina, Segmentor: An Object - Oriented Framework for Image Segmentation, *Speach and Image Understanding, Proceedings of 3rd Slovenian - German and 2nd SDRV Workshop*, pp. 331–340, April 1996.