Teleoperiranje Učnega robota preko interneta

Bojan Pregrad, Jure Muc, Mitja Truntič
Fakulteta za elektrotehniko, računalništvo in informatiko Univerza v Mariboru
Smetanova 17, 2000 Maribor, Slovenija
bojan.pregrad@email.si, jure.muc@uni-mb.si, mitja.truntic@email.si

Abstract

The paper describes a development of a teleoperating robot system via Internet. It uses a xPC target operating system developed by Mathworks as a basis for an interconnection between MATLAB software package used as control toolbox on the user PC and a robot controller server system on the other side of the internet network.

1 Uvod

Živimo v času, v katerem postaja vloga spletnih aplikacij vedno večja. Teleoperiranje je inženirska tehnična disciplina, ki daje možnost daljinskega vodenja dinamičnih procesov. Teleoperiranje lahko poteka v lokalnem (LAN) informacijskem omrežju ali pa preko javnega informacijskega omrežja. Kot javno informacijsko omrežje poznamo ethernet omrežje. Predvsem zaradi omrežnega komuniciranja se nam odpirajo različne možnosti uporabe, kot so:

- medicina (zdravnik operira pacienta na daljavo),

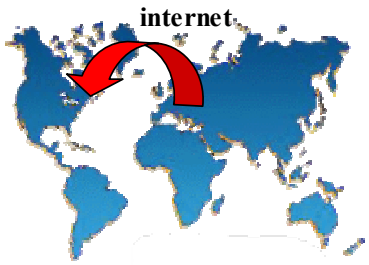
- nevarna opravila (jedrske elektrarne, deaktiviranje min, robotske podmornice…),

- vesoljske raziskave (lunarni robot) in

- virtualni laboratorij (študent izvaja aplikacije na oddaljenem robotu).

Razvita programska oprema, predstavljena v tem članku, omogoča študentom izvajanje laboratorijskih vaj na daljavo iz domačega naslonjača preko javnega informacijskega omrežja, vsak dan skozi celotno šolsko leto, 24 ur na dan.

Predstavili bomo razvito programsko opremo, ki omogoča komunikacijo med računalnikom, ki služi kot strežnik, in vmesnikom Učnega robota (slika 2).



Slika 1: Teleoperiranje preko interneta

2 Namen in cilj projekta

Realizacija projekta poteka s programskim paketom Matlab, ki je uporabljen kot osnovni študijski pripomoček na naši fakulteti (Fakulteta za elektrotehniko, računalništvo in informatiko v Mariboru). Pomembno je poudariti tudi to, da obstaja več poti za rešitev tega problema, kot so: Vlab [1] direktna aplikacija, MuMaTe [2,3] direktna aplikacija, Telegarden projekt [4]…

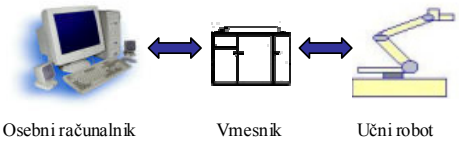
Namen projekta je povečati funkcionalnost dinamičnega sistema Učnega robota, uporabljenega kot učni pripomoček. V nadaljnje bo lahko študent opravljal laboratorijske vaje na daljavo. To pomeni, da ne bo potrebno fizično študentom in izobraževalnemu kadru prisostvovati v laboratoriju, kjer je robot, temveč bo možno voditi robota iz oddaljenega mesta. Pogoj takšnega načina vodenja je, da imamo gostujoči (uporabnikov) računalnik priključen na omrežje internet in vsebuje programski paket Matlab s pripadajočimi knjižnicami xPC Target sistema.

3 Krmiljenje Učnega robota

3.1 Izhodiščno stanje

Osebni računalnik je sposoben opravljati nalogo vodenja v notranjih koordinatah in regulacijo položaja vseh šestih osi Učnega robota. Med osebnim računalnikom in mehanskim delom robota je vmesnik, ki izvaja časovno intenzivne, vendar enostavne logične operacije. Osebni računalnik komunicira z vmesnikom Učnega robota preko paralelnih vrat.

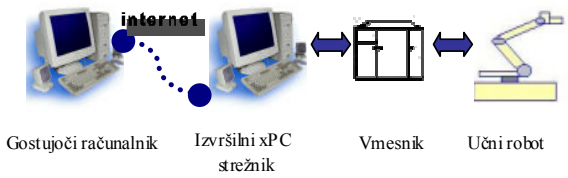
S tem je mogoča enostavna priključitev učnega robota, brez posega v strojno opremo osebnega računalnika (slika 2).



Slika 2: Krmiljenje robota znotraj laboratorija

3.2 Trenutno stanje

Obstoječo izvedbo smo nadgradili tako, da nam obstoječi računalnik služi kot izvršilni strežnik na katerem se izvaja ciljna aplikacija (slika 3).



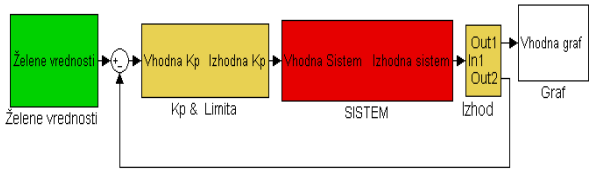
Slika 3: Krmiljenje robota preko interneta

Gostujoči računalnik nam služi za načrtovanje modela regulacijske proge, ki jo prenesemo na izvršilni strežnik preko omrežja. Komunikacija med posameznimi sklopi poteka dvosmerno, ter nam omogoča vodenje in spremljanje delovanja procesa.

Na gostujočem računalniku je implementiran programski paket Matlab 6.1 s programskim orodjem Simulink in pripadajočimi knjižnicami. Izvršilni strežnik pa vsebuje operacijski sistem xPC Target, preko katerega se direktno pošiljajo ukazi od gostujočega računalnika na vmesnik Učnega robota.

3.3 Blokovna shema

V Simulinku, kot podsistemu Matlaba, smo zgradili blokovno shemo regulacijske zanke za položajno vodenje robota (slika 4).



Slika 4: Blokovna shema v Simulinku

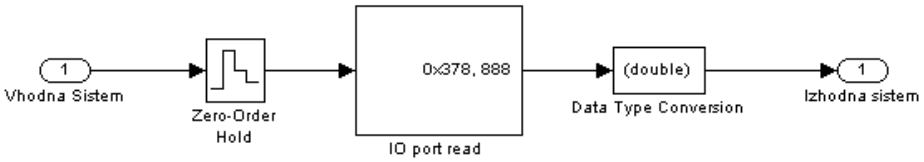
Na vhod regulacijske zanke podajamo želene vrednosti položaja posameznih osi robota. Te vrednosti primerjamo z dejanskimi vrednostmi položaja, ki jih dobimo na izhodu. Razliko položajev ustrezno ojačimo s konstanto. Na izhodu postavimo blok, s katerim lahko spremljamo odzive vseh osi Učnega robota. Odzivi so lahko prikazani grafično ali numerično.

3.4 Komunikacija

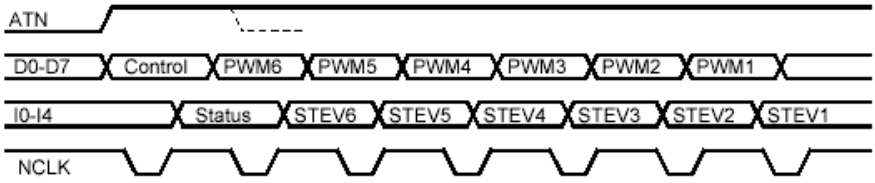
Za komunikacijo med izvršilnim strežnikom in vmesnikom Učnega robota smo zgradili lasten blok, (slika 5), ki vsebuje Matlabovo C-MEX S-funkcijo. Znotraj osnovne strukture so vključeni sklopi programa za:

- določitev vhodnih in izhodnih parametrov,
- inicializacija krmilnika,
- programska rešitev komunikacije med strežnikom in vmesnikom,
- skaliranje in omejevanje izhodne krmilne veličine in
- razširitev območja števcev.

Komunikacijska logika je del vezja, ki skrbi za pravilen dvosmeren prenos paralelnih podatkov. Na sliki 6 je prikazan komunikacijski protokol, ki ga moramo s komunikacijsko logiko realizirati.



Slika 5: Sistem s S-funkcijo



Slika 6: Komunikacijski protokol paralelne komunikacije

Komunikacija med izvršilnim strežnikom in vmesnikom za krmiljenje osi robota poteka po načinu "master-slave". Komunikacija je paralelno-serijska in dvosmerna, ki naj bi bila najboljša rešitev pri uporabljenih paralelnih vratih.

Izhodne linije strežnika so linija ATN ki določi, kdaj se prične komunikacija, podatkovne linije D0-D7, torej 8 bitni podatki in signal NCLK, ki skrbi za takt. Vhodne linije strežnika so linije I0-I4, ki vsebujejo stanje števcev položaja posameznih osi. Komunikacija se prične, ko izvršilni strežnik postavi ATN signal iz logičnega nivoja 0 na logični nivo 1. Pred tem postavi izvršilni strežnik na podatkovne izhode kontrolni register. Z prehodom NCLK iz "1" na "0" dajemo na podatkovne izhode krmilne vrednosti za delovanje motorjev. Trenutek veljavnosti krmilnih podatkov se nakaže s prehodom signala NCLK iz "1" na "0". V tem trenutku lahko vmesnik (Xilinx-ovo vezje) prebere podatkovne linije. Ob prehodu NCLK iz "0" na "1" vmesnik v Xilinx-ovem vezju nastavi statusni register vmesnika na vhodne linije paralelnih vrat izvršilnega strežnika. Pred naslednjem prehodu signala NCLK iz "1" na "0" postavi izvršilni strežnik na izhodne linije krmilni PWM signal šeste osi. Ob prehodu signala NCLK iz "0" na "1" Xilinx-ov vmesnik postavi vsebino števca inkrementalnega dajalnika šeste osi na vhode I0-I4. Po takšnem načinu se prenesejo informacije za vse preostale osi. Komunikacijo lahko prekinemo s spustom ATN linije iz "1" na "0".

3.5 Vodenje Učnega robota z xPC operacijskim sistemom

xPC Target je proizvod podjetja MathWorks za uporabo zunanjega načina »Extended mode« delovanja. Gre za izvorno-ciljno »host-target« rešitev namenjeno hitri izgradnji, razvijanju in testiranju prototipov. V tem okolju uporabljamo svoje PC namizje kot gostujoči računalnik s programskim paketom MATLAB 6.1, Simulink za kreiranje modelov, pogon simulacije in generiranje izvršne kode. Izvršilni strežnik, deluje kot ciljni »target« PC, kjer se generirana koda izvaja v realnem času in krmili vmesnik Učnega robota.

Možnosti xPC Target operacijskega sistema so:

- ustvarjanje in izgradnja ciljnih (target) aplikacij na gostujočem (host) računalniku,
- izvršitev in kontrola ciljnih aplikacij na izvršilnem strežniku s pridobitvijo signala in nastavitvijo parametrov in
- možnost vodenja aplikacije preko spletnega pregledovalnika.

3.6 xPC spletna stran

Matlab nam daje možnost vodenja ciljne aplikacije preko spletnega pregledovalnika. Osnovna stran pregledovalnika je podana že od podjetja Mathworks. Gostujoči računalnik in izvršilni strežnik je potrebno povezati preko mreže. TCP/IP komunikacijski protokol na jedru xPC ciljnega sistema nam podpira samo eno sočasno povezavo, saj je naloga vodenje v realnem času.

Preden se spletni pregledovalnik poveže na izvršilni strežnik mora biti aplikacija na strežniku naložena. Pregledovalnik mora vsebovati Java Script in oblikovne knjižnice.

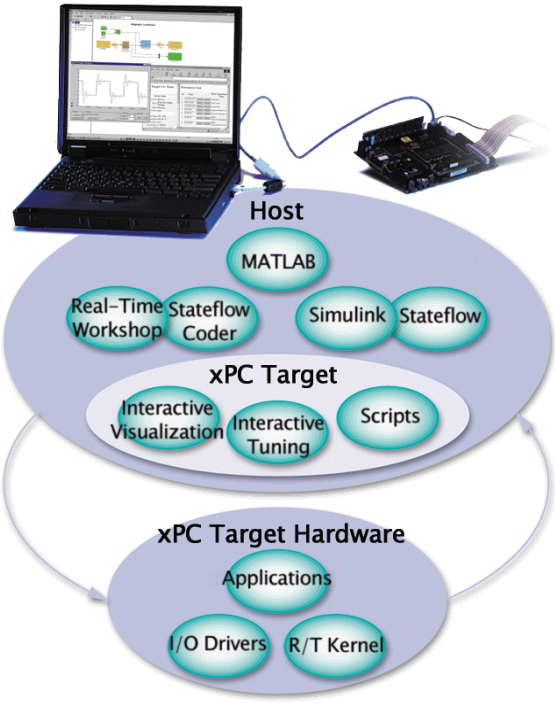
3.7 Zaključek

Pri tem projektu smo realizirali teleoperiranje Učnega robota preko gostujočega in izvršilnega strežnika , kateri je s paralelnimi vrati priključen na vmesnik Učnega robota. Težave s katerimi smo se srečevali so bile predvsem pri pisanju programa, za komunikacijo strežnika in vmesnika robota. Izdelati je bilo potrebno komunikacijski protokol, skaliranje in omejitev krmilne veličine, ter razširitev območja števcev položaja. Trenutno lahko robota vodimo v notranjih koordinatah.

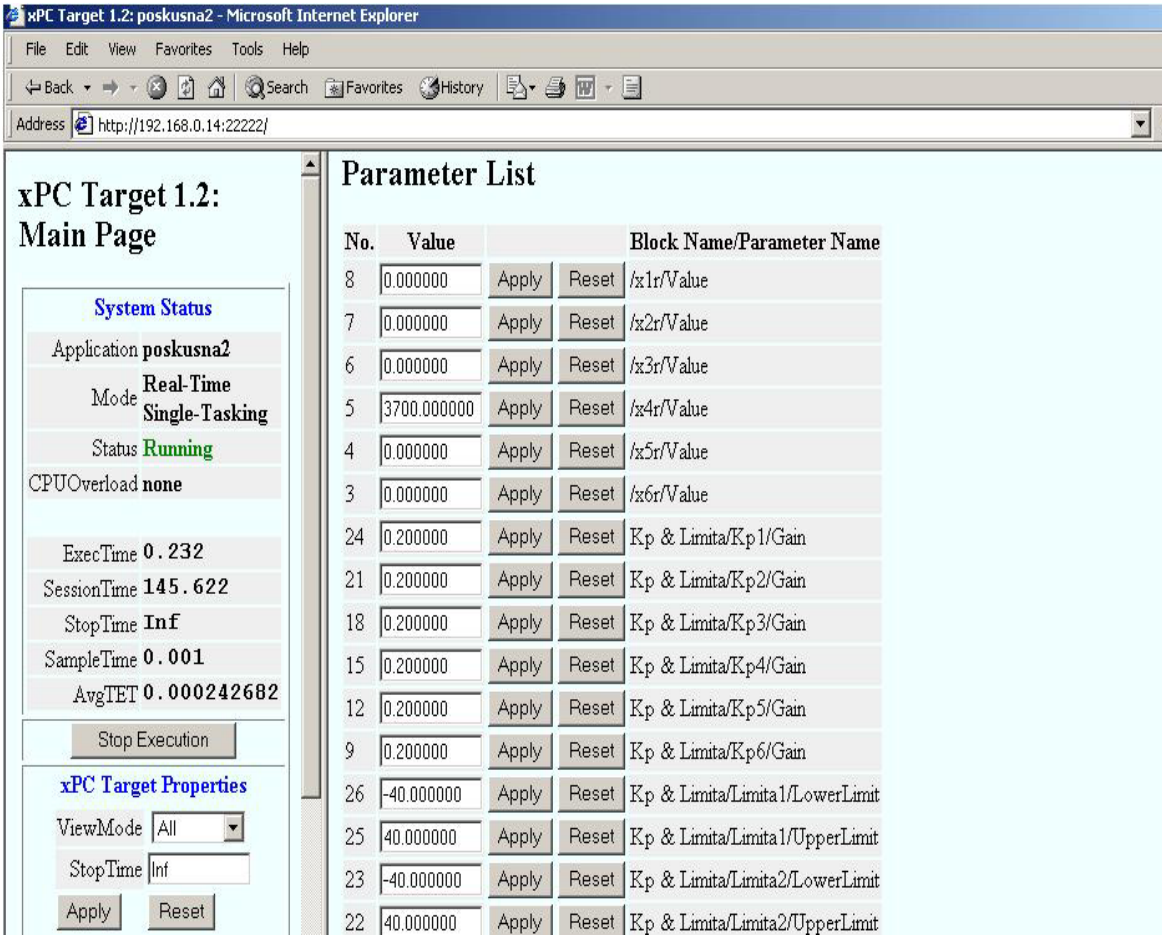
Želja v prihodnosti je povezava obstoječe programske opreme z Vlab strežnikom:

- Priključitev na vmesnik ***xPC Proxy***,
- priključitev xPC Proxy na Vlab strežnik,
- povezava Vlab strežnika z VRML modelom robota,
- vključitev detekcije kolizije v VRML model in
- vzpostavitev povezave VRML, Vlab, xPC Proxy, xPC Target izvršilni strežnik.

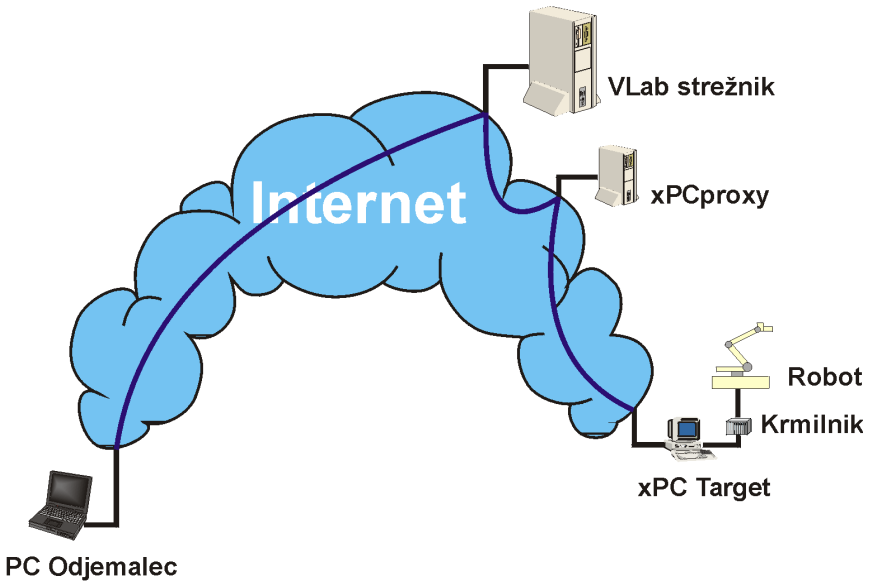
Vlab strežnik je implementiran v smislu relacije **odjemalec-strežnik**.
VLab odjemalec je dinamična spletna aplikacija, ki poteka na odjemalcu in s katero uporabnik upravlja simulacijo in vodenje realnega robotskega sistem.
VLab strežnik je platforma s programsko opremo, ki streže vsebino odjemalcu, hkrati pa vsebuje bazo podatkov za projektno delo, verifikacijo uporabnika in komunikacijo z izvršilnimi strežniki (ang. executive servers). Strežnik vsebuje tudi urnik za delo s posameznimi delovnimi celicami.



Slika 7: xPC Target sistem



Slika 8: Prikaz xPC spletne strani



Slika 9: Zasnova sistema Vlab

xPCproxy je spletna aplikacija, ki predstavlja most med VLab sistemom in xPC Target izvršilnim strežnikom. Aplikacijo lahko razdelimo na izvršilni in administratorski del. Izvršilni del skrbi za komunikacijo med VLab-om in xPC Target-om, administratorski pa za vnaprejšnjo uskladitev podatkov, ki se izmenjujejo med navedenima sistemoma. Sistem je izdelan predvsem s tehnologijo .NET in sicer s programskim jezikom C#, razen knjižnica za komunikacijo z xPC Target-om, ki je napisana v programskem jeziku Visual Basic 6.0.

4. Literatura

[1] ŠAFARIČ, Riko, DEBEVC, Matjaž, PARKIN, Rob M., URAN, Suzana. Telerobotics experiments via Internet. IEEE trans. ind. electron. (1982), Apr. 2001, vol. 48, no. 2, str. 424-431.

[2] ŠAFARIČ, Riko, CALKIN, D. W., PARKIN, R. M., CZARNECKI, C. A. Virtual environment for telerobotics. Integr. comput.-aided eng., Apr. 2001, vol. 8, no. 2, str. 95-104.

[3] ŠINJUR, Smiljan, ŠAFARIČ, Riko, ŽALIK, Borut. A JAVA collision detection for a WEB-based robot. V: TAR, J. K. (ur.). 11th International Workshop on Robotics in Alpe-Adria-Danube Region RAAD 2002, Balatonfüred, Hungary, June 30 - July 2, 2002. Proceedings. Budapest: Budapest Polytechnic, 2002, str. 223-228.

[4] http:\\www.sciencedirect.com\science\ journal\09574158.

[5] http://www.mathworks.com

[6] MATLAB/Simulink Help