Simulacija minobacača

Seminarski rad iz kolegija “Interaktivni simulacijski sustavi”

Renato Majer, Dalen Grdić, David Kovačević, Mario Petek  
20.01.2023.

Djelovođa: izv. Prof. dr. sc. Siniša Popović

***Sažetak* – Kroz interaktivnu simulaciju demonstrirana je okolina u kojoj vojnici sa stacionarne pozicije pomoću minobacača gađaju neprijateljske tenkove koji pristižu sa različitih strana i nalaze se na različitim udaljenostima od samih vojnika. Vojnici mogu okretati minobacač prema lijevo i prema desno te isto tako mijenjati kut pod kojim se ispaljuju mine. S druge strane, tenkovi ispaljuju projektile prema vojnicima, pa je vojnicima u cilju čim prije uništiti neprijateljske tenkove.**

# Uvod

Vojne simulacije su jako bitne za uvježbavanje vojnih taktika ili provođenje vojnih vježbi bez stvarne opasnosti za njihove sudionike uz značajno smanjene troškove u odnosu na trošak stvarnog provođenja iste.

Okolina ove vojne simulacije je utvrđena pozicija vojnika s minobacačem s pogledom na otvoreno polje okruženo drvećem i raslinjem gdje se očekuje nailazak neprijateljskih tenkova. Tenkovi nailaze s obje strane i dobivaju položaj utvrđenih vojnika nakon čega se okreću prema poziciji minobacača i ulaze u konflikt.

Cilj simulacije je uvježbati korištenje minobacača u mogućoj stvarnoj situaciji gdje je njihova pozicija kompromitirana. Zadatak vojnika je uništiti neprijatelja korištenjem minobacača prije nego neprijatelj pogodi njihovu utvrđenu poziciju. Minobacač se može okretati strelicama lijevo i desno, kut gađanja strelicama gore i dolje, a mina se ispaljuje lijevim klikom miša.

Zanimljive značajke ove simulacije su modularnost same simulacije. Stvaranje i pucanje tenkova može se ostvariti ručno na stisak pripadajuće tipke ili kontinuirano čime se može kontrolirati opterećenje i opasnost za vojnike gdje je kontinuirano stvaranje i pucanje tenkova određeno nasumičnim vremenom. Dodatno, pucanje je ostvareno nasumičnom preciznošću kako bi simulirali stresnu situaciju bliskog pogotka u utvrđenu poziciju. Simulacija se dodatno može ponovno pokrenuti po potrebi, a sve mogućnosti prikazane su u kutu ekrana.

# Uloge pojedinih članova tima

U sveukupnim poslovima na izradi seminarskog rada te pisanju ovog izvješća, članovi tima sudjelovali su na sljedeći način:

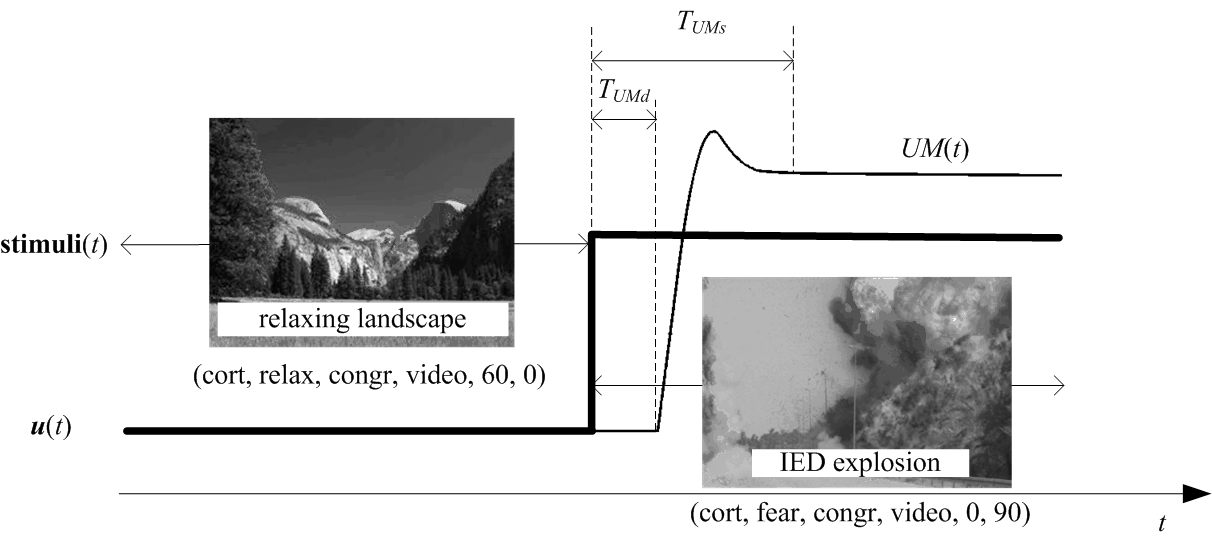
* Ime Prezime – <Voditelj tima napisati prvo svoje ime i prezime, te poslove koje je obavio voditelj. Nakon toga, treba u zagradi spomenuti specifične sekcije/podsekcije ovog izvješća kojima je voditelj dao doprinos>
* Ime Prezime – <Voditelj tima napisati ime i prezime, te poslove koje je obavila ova osoba. Nakon toga, treba u zagradi spomenuti specifične sekcije/podsekcije ovog izvješća kojima je ova osoba dala doprinos>
* Ime Prezime – <Voditelj tima napisati ime i prezime, te poslove koje je obavila ova osoba. Nakon toga, treba u zagradi spomenuti specifične sekcije/podsekcije ovog izvješća kojima je ova osoba dala doprinos>
* ... <Voditelj tima dopisati sve ostale članove tima i poslove koje su obavili, na isti način kao za prethodne>.

# Razvojno okruženje

## Unity

Implementacija simulacije ostvarena je u programu Unity, popularnom razvojnom okruženju za razvoj video igara baziran na programskom jeziku C#. Unity omogućuje stvaranje fizike, 3D renderiranje, detekciju sudara i druge stvari koje našu simulaciju čine funkcionalnom. Alat je pogodan jer omogućuje njihovo direktno korištenje bez potrebe za programiranjem čime je focus prebačen na samu implementaciju.

Unity ima vizualni uređivač kojim u par klikova možemo stvoriti osnovne objekte, urediti, ali i upravljati njihovim svojstvima te promatrati ponašanje u prostoru. Programski jezik zaslužan za rukovanje kodom, logikom i drugim klasama jedinstvenih za Unity je C#. Sve navedeno omogućuje programerima raznih iskustva i vještina lakoću rukovanja ovim alatom. Unity također ima vlastiti forum i „***Assets store***“, mjesto gdje programeri prenose svoje kreacije i čine ih dostupnim široj zajednici zbog čega je izabran za izradu ove simulacije.



Slika 1. Slika u primjeru je siva, ali naravno da u seminarskom radu mogu biti i slike u boji. Pripaziti da prilikom ispisa slika u boji na crno-bijelom pisaču slike i dalje ostanu razumljive čitatelju. Slike treba pozicionirati nakon što se sliku prvi puta spominje u tekstu (najbolje na istoj stranici, ili na stranici iza spominjanja u tekstu). Ako ste sliku preuzeli iz nekog izvora (tj. niste ju sami napravili), trebate obavezno na kraju naslova slike staviti broj izvora u uglatim zagradama iz kojeg ste sliku preuzeli, npr. „preuzeto iz [1].“

Ako se u seminarskom radu opisuje vlastiti praktični rad, onda u ovoj i sljedećim specifičnim sekcijama seminarskog rada opisujete ukratko što su drugi napravili a relevantno je za Vašu temu (koncizni pregled literature ili karakteristika infrastrukture koju koristite u radu u segmentima koji su posebno relevantni za Vaš praktični rad), te zatim Vaše metode (kako ste nešto napravili – npr. neke od tema kao što su Vši modeli, arhitekture, oblikovanja, implementacije itd.), rezultate (što ste dobili) i diskusiju (npr. što rezultati znače, kako kotiraju spram drugih sličnih radova koji su se bavili srodnom temom, koja su ograničenja Vašeg rješenja, da li bi rezultati mogli vrijediti u općenitijem kontekstu od onoga u kojemu ste ih dobili…). Eventualne izvatke programskog koda, pseudokod algoritama itd. treba prikazati kao slike. Posebno treba obratiti pažnju da seminarski rad nije samo opis funkcionalnosti i načina korištenja ostvarenog programskog rješenja (jasno da to svakako treba imati), već je potrebno objasniti ključne stvari za razumijevanje kako je rješenje s tehničke strane oblikovano i razvijeno, što su njegove ključne sastavnice te kako je ostvarena njihova interakcija u realizaciji cjelokupnog programskog rješenja. Drugim riječima, izvješće nije samo dokumentacija za potencijalnog korisnika Vašeg rješenja, već prije svega tehničko-inženjerska dokumentacija da bi inženjerski obrazovana osoba mogla razumjeti kako je rješenje ostvareno. Ključne sastavnice sa strane oblikovanja i razvoja rješenja mogu se, primjerice, odnositi na razrede u objektno orijentiranom modelu, procese/dretve ako postoji paralelizam, eventualne specifične algoritme kojima su realizirani neki ključni dijelovi rješenja, strukture ulaznih/izlaznih podataka/datoteka, model eventualne baze podataka, organizaciju programskog koda u datoteke itd. U tom smislu, nije nimalo neobično, štoviše očekivano je, da se ovdje pojavljuju neki tehnički dijagrami i opisi, kao što su npr. dijagrami razreda, strojevi stanja, dijagrami toka, interakcijski UML dijagrami, opisi/strukture komunikacijskih poruka, modeli entiteta i veza, pseudokodovi algoritama itd.

## Glavni elementi

Unity svoje prizore naziva **scene**. To je prizor na kojem vidimo izgled naše trenutno ostvarene okoline. Scena sadrži **objekte** koje želimo da se u njoj nalaze i upravljaju tokom njenog izvođenja, a objekte je moguće razmještati. Svakom objektu i njegovim svojstvima može se pristupiti ugrađenim **inspektorom**.

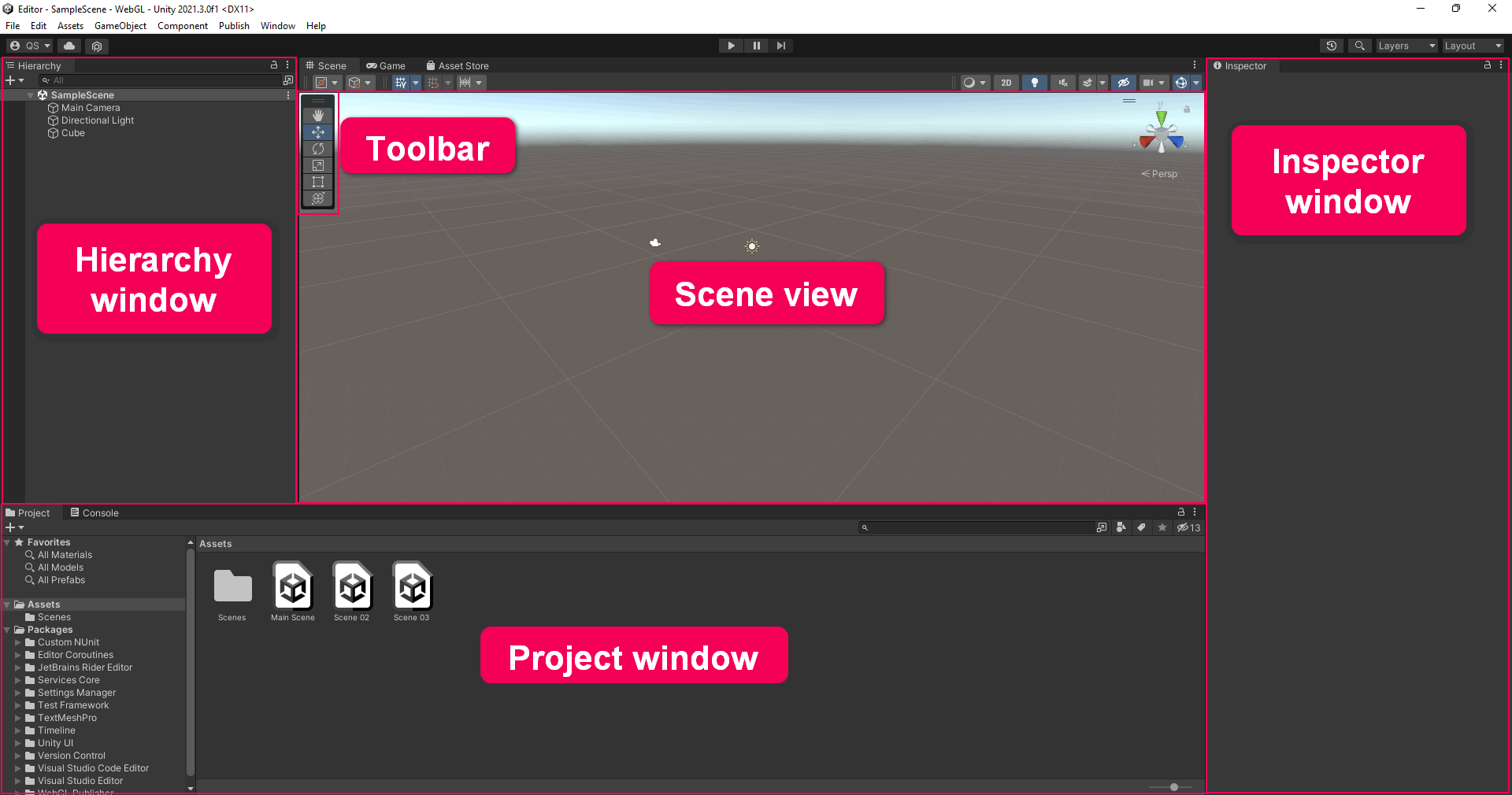
Scena također sadrži **kameru**, koja stavlja korisnika u scenu te izvor svjetlosti koja služi za osvjetljavanje scene i objekte svjetlošću iz tog smjera.

Prozori **Game** i **Scene** nam prikazuju prizor iz perspektive koju želimo da korisnik vid i promatra, to jest, trenutno stanje prizora i omogućuje stvaranje promjena koje postaju vidljive iz perspective korisnika za vrijeme njenog izvođenja.

## Objekti i modeli

Prizor scene bez objekta je ekran bez prikaza što znači da i sama kamera ima ulogu objekta u sceni što čini objekte nezaobilaznim dijelom u procesu implementacije. Kada objekt ili grupa objekata ima nama semantičko značenje ono nazivamo model. Model nam omogućuje jednostavniju implementaciju ponašanja i manipulaciju objekata u sceni grupiranjem objekata u logičku skupinu.

Objekt se sastoji od komponenti, a svaki objekt ima komponentu transformacije[] koja sadrži podatke o poziciji, rotaciji i skaliranju objekta, a opcionalno može imati i komponente važne za ovu simulaciju poput skripti[], krutih tijela[], sudarače[], zvučne izvore[], animatore[], ali i mnoge druge.



Slika 1. Grafičko sučelje alata Unity. Preuzeto iz []

## <Prva podsekcija>

Ovo je specifična podsekcija koja ovisi o temi koja se obrađuje. Stoga joj treba dati odgovarajući naslov i u tijelu dati odgovarajući tekst. Naravno, tekst seminarskog rada ne mora nužno sadržavati podsekcije.

# Programski kod

Da bi upravljali simulacijom, određeni objekti moraju biti programirani. Objekti znaju svoj položaj i oblik u prizoru, ali ne znaju se kretati ili pozicionirati za vrijeme izvođenja bez postojanja programskog koda koji upravlja njihovim svojstvima. Zbog toga su objekti kojima želimo manipulirati i mijenjati njihova svojstva tijekom izvođenja upravljani skriptama pisanim u programskom jeziku C#.

## MonoBehaviour

Sve klase ove simulacije nasljeđuju baznu klasu **MonoBehaviour** za implementaciju posebnih funkcija.

Funkcija **Start** se poziva na početku pokretanja skripte i koristi se za incijalizaciju varijabli potrebne za ispravno izvođenje ostalih funkcija.

**Update** je korisna funkcija koja se poziva nakon svake prikazane sličice i najčešće se koristi za implementaciju kretanja objekta u prizoru.

Funkcija **Instantiate** i **Destroy** koriste se za stvaranje i uništavanje objekata u prizoru i najčešće su implementirane kroz korištenje funkcije **OnCollisionEnter** kojom pratimo sudare različitih objekata u simulaciji.

Osim ovih funkcija koristi se i **Invoke** koja omogućuje pozivanje funkcija nakon određenog vremenskog perioda što nam daje opciju za tempiranje određenih događaja u prizoru.

## MortarShell

Ovom klasom modeliramo let mine iz minobacača. Mina ima komponentu **krutog tijela** kojom možemo upravljati masom, ali i površinskim i kutnim **otporom** što omogućuje realističan **kosi hitac** minobacačem. Također sadrži komponentu **sudarača** koji na sudar s terenom stvara eksploziju s kraterom na mjestu udara ili običnu eksploziju u slučaju drugih objekata. Ukoliko se nađe ispod razine terena, objekt mine će se uništiti funkcijom **Destroy**.

## MortarAction

Da bi mogli izbaciti minu iz minobacača, moramo implementirati funkcije koje će stvoriti minu i izbaciti je u smjeru gađanja. Ova klasa koristi metodu **Fire** koja se poziva iz vansjke skripte na stisak gumba za pucanje, a koja stvara vizualni efekt izbacivanja mine, a zatim funkciju **FireMortarRound** koja stvara objekt mine te se na objekt mine primjenjuje sila izbačaja.

## TankAction

Slično klasi **MortarAction** ovom klasom definirana je jednostavna funkcionalnost kretanja neprijatelja. Neprijatelji se kreće prema sredini, a zatim prema utvrđenoj poziciji. Neprijatelj staje na nasumičnoj poziciji gdje predstavlja opasnost vojnicima.

## TankShooting

Ova klasa implementira logiku ispaljivanja projektila u nasumičnom smjeru prema vojnicima kako bi simulirali promašaje. Dodatno, projektili se mogu i ručno ispaljivati na stisak gumba, ali i podesiti da pogodak projektila u vojnike yavr[ava simulaciju.

## SpawnerScript

Da bi upravljali tokom simulacije potrebno je implementirati stvaranje neprijatelja, utvrđene pozicije i slično. Zato ova skripta omogućuje ručno ili periodičko stvaranje neprijatelja kako bi kontrolirali opterećenje i stres nad vojnicima i resetiranje prizora na stisak tipke.

# Pokretanje simulacije

Simulacija se pokreće dvostrukim klikom miša na izvršnu datoteku **„MortarSimulation.exe“** koja se nalazi u projektnoj mapi **Build** čime započinje simulacija.

# Zaključak

Cilj simulacije je postaviti vojnike u stresnu situaciju koja bi ih potencijalno mogli zateći. Osim toga, za vojnike je važno da razviju motoriku upravljanja minobacačem kako bi bili što spremniji u slučaju stvarnog konflikta. Iako današnje 3D simulacije mogu prikazati izrazito realistične situacije, da bi postigli puno bolje efekte, ova simulacija trebala bi se implementirati za virtualnu realnost, gdje će simulacija biti stvarnija i uzrokovati realniji stres. Virtualna realnost omogućila bi korištenje raznih senzora pokreta tijela, ponajviše ruku te uz stvarni model minobacača s nekim oblikom trzaja.

# Literatura

[1] Ovo je format stavki u literaturi. Ispod možete naći primjer informacija koje treba popisati u literaturi ako se radi o knjizi [2], članku u časopisu [3], članku u zborniku konferencije [4], patentu [5], tehničkom izvješću [6], elektroničkoj knjizi [7], te elektroničkim priručnicima i izvješćima [8]. Ako neke informacije o određenom izvoru ne možete naći, kao npr. datum objave elektroničke knjige, izostavite ih. Ako u pretrazi literature naiđete da kao autori djela nisu navedeni specifični ljudi, već firme, organizacije, ili institucije, onda tako navedite djelo i u vašem popisu literature. **U literaturi treba navesti barem 3 stavke, te svaka stavka iz literature treba ujedno biti citirana u tekstu vašeg seminara.**

[2] S. M. Sze, Physics of Semiconductor Devices, 2nd Edn., Wiley, New York, 1981.

[3] D. S. Lee and J. G. Fossum, “Energy Band Distortion in Highly Doped Silicon”, IEEE Transactions on Electronic Devices, vol. 30, p. 626, 1983.

[4] D. B. Payne and J. R. Stern, “Wavelength-switched passively coupled single-mode optical network,” in Proceedings of *IOOC-ECOC*, 1985, pp. 585–590.

[5] G. Brandli and M. Dick, “Alternating current fed power supply,” U.S. Patent 4 084 217, Nov. 4, 1978.

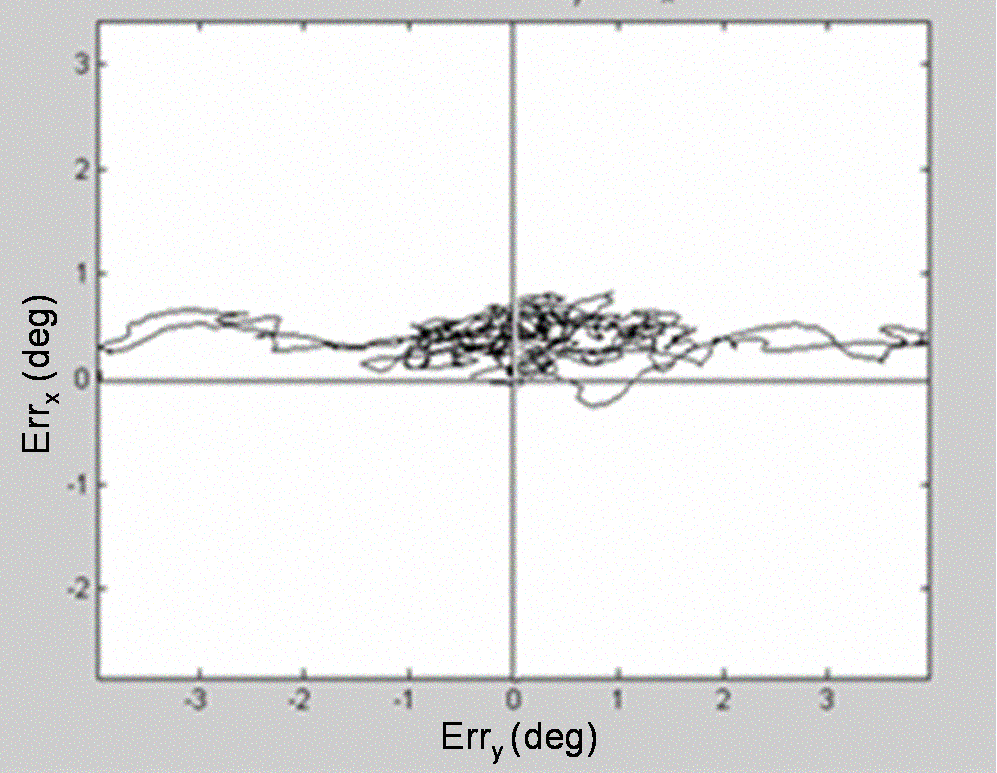
[6] E. E. Reber, R. L. Mitchell, and C. J. Carter, “Oxygen absorption in the Earth’s atmosphere,” Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (4230-46)-3, Nov. 1968.

[7] J. Jones. (10. svibnja 1991.). Networks. (2nd ed.) [Online]. http://www.atm.com. Datum pristupa: 14. veljače 1999.

[8] S. L. Talleen. (lipanj 1996.). The Intranet Architecture: Managing information in the new paradigm. Amdahl Corp., CA. [Online]. http://www.amdahl.com/doc/products/bsg/  
intra/infra/html. Datum pristupa: 8. lipnja 1998.

# Dodatak

Ovdje je moguće nanizati koliko god je potrebno slika koje su interesantne za uključiti, ali ih ima previše da sve budu uvrštene u tekst rada. Naime u količini većoj od predviđene u uputama, stavljanje slika u tekst bi moglo narušiti čitljivost rada. Kao i u tekstu seminarskog rada, svaka slika u dodatku treba imati naslov i referirati izvor iz kojeg je preuzeta (ako jest preuzeta).



Slika 2. Ovo je primjer slike u dodatku.