Sustav za definiranje i vizualizaciju konstelacija bespilotnih letjelica

Bogešić, Marin

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Marin Bogešić

Sustav za definiranje i vizualizaciju konstelacija bespilotnih letjelica

ZAVRŠNI RAD

Varaždin, 2022.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Marin Bogešić

Matični broj: 00161374415

Studij: Informacijski sustavi

SUSTAV ZA DEFINIRANJE I VIZUALIZACIJU KONSTELACIJA BESPILOTNIH LETJELICA

ZAVRŠNI RAD

Mentor/Mentorica:

Doc. dr. sc. Boris Tomaš

Varaždin, studeni 2022.

Marin Bogešić

Izjava o izvornosti

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor/Autorica potvrdio/potvrdila prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

U ovome radu opisuje se način funkcioniranja Sustava za definiranje konstelacija bespilotnih letjelica. U početku se teorijski opisuje dron, bespilotna letjelica te moguće razlike između pojma drona i ostalih bespilotnih letjelica. Nabrajaju se vrste dronova i njihove svrhe u modernom svijetu te se prikazuju neki primjeri visoko-tehnološki razvijenih bespilotnih letjelica. Drugi dio rada obuhvaća detaljni opis samog sustava koji je izrađen u Unity softveru. Opisuju se same funkcionalnosti sustava po SRS standardu te se svaka funkcionalnost vizualno prikazuje uz određeni dijagram te se opisuje način na koji proces funkcionira. Uz to prikazuju se dijelovi programa kojima se treba posebno obratiti pažnja. To su dijelovi programa koji opisuju krucijalne dijelove sustava, najviše dijelove koji opisuju način rada i funkcioniranja senzora kod pojedinog drona te način izbjegavanja kolizije sa bilo kojim dronom unutar simulacije. Ističe se kako je ova simulacija prva verzija sustava te sam sustav nije u svakoj situaciji u potpunosti točan. S druge strane, sustav svakako prikazuje točne rezultate u većini slučajeva, s obzirom na potrebu izbjegavanja kolizija dronova u letu.

Ključne riječi: dron; Unity; SRS; Funkcionalnosti; Senzori; Kolizije; Verzija sustava; Susta

Sadržaj

1. Uvod

U današnjem svijetu pojam bespilotnih letjelica gleda se kao proizvod sa puno potencijala i novih mogućnosti. To je tehnologija koja na prvi pogled otvara veliki broj vrata za inovacije, olakšanje i poboljšanje kvalitete života za njihove korisnike. U nekim slučajevima može se gledati čak kao i alat za spašavanje života unutar službа spašavanja. S druge strane, neozbiljno i neodgovorno korištenje dronova može donijeti velike posljedice. To mogu biti posljedice poput privremenog zaustavljanja letova kao što se dogodilo 2019. godine u Londonu koje je neposredno nanijelo multi milijunašku štetu avio-kompanijama [1]. Gledajući na sve dobre i loše strane ovog tehnološkog napretka, jedino što sigurno možemo zaključiti je da se mukotrpne pravne regulacije tek počinju pravilno razvijati u usporedbi sa nezaustavljivim razvojem tehnologija.

Dok se sklapaju dogovori o pravilnom i sigurnom korištenju dronova, tehnološki napredak nastavlja rasti te se proizvodi nastavljaju razvijati. Među tim naprecima dolazi se do novog pojma 'roja' dronova. Skupina dronova koji imaju određeni sklad i formaciju leta te se razlikuju od skupine dronova kod koje svaki dron autonomno i samostalno leti bez obzira na druge dronove unutar svoje skupine. Ovaj rad djelom služi kao dokumentacija za sustav koji se koristi za definiranje konstelacija bespilotnih letjelica izrađen od strane autora rada. Sustav se koristi kako bi se prikazala simulacija leta određenog broja bespilotnih letjelica bez međusobnih kolizija.

Glavna problematika roja dronova su potencijalne kolizije i načini rješavanja tih problema. Ovaj rad opisuje jedan od načina rješavanja problema uz određeni broj dronova unutar simulacije.

2. Metode i tehnike rada

Tema završnog rada je kreirati sustav koji će se koristiti za definiranje i postavljanje proizvoljnih konstelacija određenog broja dronova koji zajedno čine roj, u svrhu simuliranja i prikaza načina na koji će roj letjeti i dovršiti svoju putanju bez međusobne kolizije.

Za potrebe završnog rada koristi se Unity platforma za izradbu simulacije u kojoj će postojati mogućnosti definiranja konstelacija, tj. formacija određenog broja dronova unutar određenog vremena i prostora. Dodatne mogućnosti koje se ubrajaju u postavke simulacije su postavljanje određenog modela drona, broj dronova koji lete i minimalno ili maksimalno vrijeme koje se postavlja kako bi se izmjerilo koju daljinu roj može preći u postavljenom vremenskom razdoblju.

Primarno se simulacija dijeli na tri dijela aktivnosti roja. Polijetanje, let i slijetanje roja. U fazi polijetanja dronovi se uzdižu na određene točke te ukoliko su svi dronovi spremni, kreću na sljedeću fazu; kontinuirani let. Za vrijeme leta dronovi provjeravaju udaljenost jedan među drugima te se izbjegavaju moguće kolizije među dronovima. Nakon uspješnog prelaska faze leta te dolaskom do odredišta se nastavlja na sljedeću fazu; faza slijetanja. U fazi slijetanja dronovi pojedinačno biraju točke slijetanja te, sinkroniziranim pokretima, zajedno slijeću na odabranu krajnju točku slijetanja.

Slika 1: Unity Logo [10]

3. Razrada teme

Kako bismo napravili sustav za definiranje konstelacija dronova, moramo prvo definirati pod što se uopće smatra dron. Razlike između UAV (*eng. Unmanned Aerial Vehicle*) i drona. Nakon toga će se razraditi ideja skupine dronova kao roja te što ih čini rojem, a ne skupinom dronova koje samo simultano lete jedan pored drugoga. Nakon toga ćemo opisati sam Sustav za definiranje Konstelacije Bespilotnih Letjelica, njen način rada i aktivnosti unutar programa uz algoritam izbjegavanja kolizija bespilotnih letjelica unutar simulacije.

3.1. Bespilotne letjelice

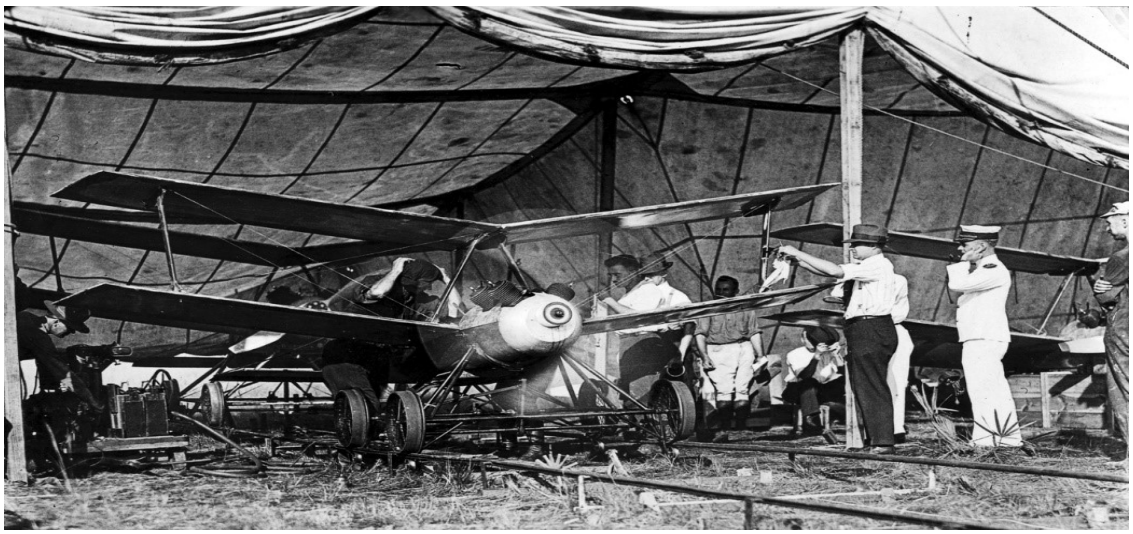
Za bespilotnu letjelicu se podrazumijeva da je leteći objekt koji se upravlja daljinskim upravljačem *(eng. Remotely Piloted Aircraft System)* ili samostalno leti uz pomoć već isplaniranog plana leta pod uvjetom da im je taj softverski plan leta već ugrađen u sustav *(eng. Unmanned Aircraft System).* Također se smatra kako bespilotna letjelica ne može ispravno i sigurno funkcionirati ukoliko nema ugrađene senzore i/ili globalni sustav pozicioniranja (GPS). Postoje razne vrste bespilotnih letjelica budući da se svaka koristi za specifičnu ulogu; od poljoprivrednih poslova do letjelica koje se koriste u vojne svrhe. U nastavku će se proći povijest bespilotnih letjelica i nj razvoja te će se nabrojati vrste letjelica i njihove svrhe korištenja.

Slika 2: Bespilotna letjelica [11]

3.1.1. Povijest

Prve letjelice koje su bile upravljane bez prisustva pilota isto tako su bile razvijene za vojne svrhe u Engleskoj i SAD-u tokom Prvog Svjetskog Rata. Naime, nijedna od tih dviju država nije iskoristila te prve letjelice za vojne operacije u tom razdoblju, unatoč dobrim rezultatima na testiranjima. Jedna od prvih takvih letjelica, točnije UAT (*eng. Unmanned Aerial Torpedo*), se zvao *Kettering Bug*. Tadašnji eksperimentalni 'Krstareći Projektil' *(eng. Cruise missiles*) bio je sposoban letjeti brzinom od 80 kilometara na sat u daljinu od 121 kilometara. Prvi UAV koji su se koristili u vojne svrhe su zapravo bili dronovi na daljinsko upravljanje koji su se testirali neposredno nakon Drugog Svjetskog Rata. Bespilotne letjelice koje su u sebi imale implementirane infra-crvene kamere, kontrolnu stanicu i druge osnovne senzore prvi put su bile korištene od strane Izraelskih Obrambenih Snaga 1980-ih godina. Vidjevši njihov uspjeh koji se iskazao zbog glavnih karakteristika takvih letjelica; da su malene i tihe, Američke vojne snage su kupili ranije modele od Izraelske vojske i time ih nastavili sami razvijati.

UAV koje su se koristile za izviđanje prvi put su bile 'ozbiljno' korištene tokom Vijetnamskog rata. U tom razdoblju bespilotne letjelice su dobile nove uloge, poput mamca u strateškim borbama, lansiranje projektila na fiksne ciljeve i bacanje letaka za psihološke operacije. Nakon 11. Rujna (Napad na Blizance), Sjedinjene Američke Države su značajno povećale svoju upotrebu bespilotnih letjelica u svrhu nadzora i izviđanja područja i terena koji nisu sigurni za vojnike. Osim nadzora, danas se vojne bespilotne letjelice najviše koriste za eliminiranje osumnjičenih militantna i državnih neprijatelja. Opće je poznato i etičko pitanje korištenja ove vrste oružja, s obzirom da korištenje projektila bespilotnih letjelica često rezultira smrtnim slučajevima civila.

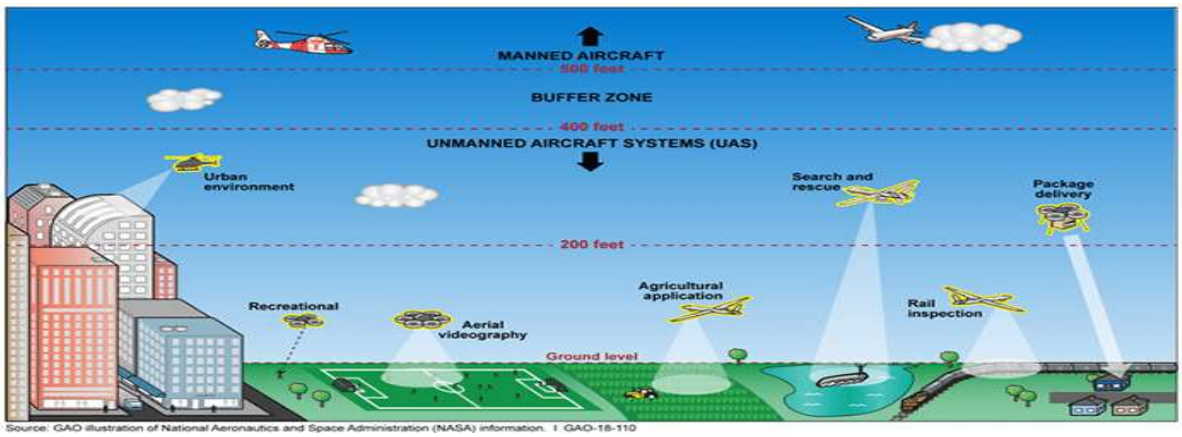


Slika 3: Kettering bespilotna letjelica [12]

3.1.2. Vrste i njihova svrha

Upravljanje bespilotnim letjelicama najčešće se dijeli na 4 svrhe korištenja: osobne, komercijalne, industrijske i državne svrhe. Prema tim ulogama razlikujemo 3 vrste bespilotnih letjelica koje se danas koriste: Bespilotne letjelice (*eng. Unmanned Aerial Vehicle – UAV*) i Letjelice na daljinsko upravljanje (*eng. Remotely Piloted Vehicle - RPV*) koje smo već spomenuli te Dronovi (*eng. Drones*). U nastavku ćemo opširnije proći svaku vrstu i neke od njihovih načina korištenja. U današnjem svijetu često je naziv Bespilotna letjelica, preciznije sam naziv UAV, samo sinonim za Dron. Dok je u općenitom smislu to istina, tehnički gledano postoje ozbiljne razlike između spomenutih naziva. Preciznije, možemo reći: *„Svaki UAV je dron, ali nije svaki dron UAV“* [2]. Jedna od ključnih karakteristika zbog koje se ističe razlika je ta da se smatra kako Bespilotne letjelice (*UAV*) moraju imati sposobnost autonomnog leta, za razliku od Dronova [3]. Neke od važnijih uloga u kojima se koriste Bespilotne letjelice su tzv. Zadaci opće sigurnosti:

- Kontrola požara
- Lov na uragane
- Pomoć u katastrofama i humanitarna pomoć
- Kontrola kriminala
- Potraga i spašavanje
- 3D mapiranje
- Dostava proizvoda i hrane
- Rudarstvo i poljoprivreda



Slika 4: Dronovi i njihove uloge [13]

Osim razlikovanja bespilotnih letjelica po njihovoj razini autonomnosti, bespilotne letjelice možemo razlikovati i po njihovoj standardnoj visini leta. Slijedno tome, koristi se terminologija Bespilotni Zračni Sustav (*eng. Unmanned Aerial System, UAS*) kojom definiramo cjelokupnu operativnu opremu uključujući zrakoplov, kontrolnu stanicu s koje se upravlja zrakoplovom i bežičnu podatkovnu vezu [4]. U nastavku ćemo govoriti o podjeli Bespilotnih letjelica prema visini leta, specifično letjelice koje stavljamo pod strogu, prije spomenutu, definiciju UAV (*Unmanned-Aerial Vehicle*) i RPV (*Remote-Pilloted Vehicle*). Nakon toga ćemo posebno istaknuti bespilotne letjelice poznate kao Dronovi.

3.1.3. Bespilotne letjelice prema visini leta

Bespilotne Zračne Sustave dijelimo na tri vrste prema već spomenutom kriteriju visini leta:

- HALE UAS (*eng. High-Altitude Long- Endurance*)
- MALE UAS (*eng. Medium-Altitude Long-Endurance*)
- Small UAS

Duga izdržljivost na velikoj visini (*HALE*) se koristi kako bi se opisalo zračno vozilo koje funkcionira optimalno na visinama do 18 km, ali znatno sporijom brzinom od ostalih bespilotnih letjelica. *HALE* bespilotne letjelice često imaju karakteristike prijašnje definiranih *UAV* i *RPV* bespilotnih letjelica. Razlog tome je što kod većinu *HALE* letjelica se koristi manualno upravljanje pri uzletu i sletu *HALE* bespilotnih letjelica. Zbog visokih kriterija koje su predstavljeni ovakvim letjelicama, uključujući zahtjev da se ostvari let na velikim udaljenostima u okruženju s razrijeđenim zrakom i niskim temperaturama, osmislili su se mnoga inovativna rješenja. Neki od tih rješenja uključuju korištenje neiscrpnih izvora energije (solarni paneli i sl.), povećanje gustoće energije za pohranu energije (primjer: čvrsti vodik) i slično. [5]

Primjer takve letjelice je *PathFinder*, tj. *PathFinder PLUS*. Spomenute letjelice su bile dio evolucijske serije bespilotnih letjelica s pogonom na solarne i gorive ćelije. Letjelice su bile razvijene u sklopu NASA-inog programa za istraživanje okoliša i tehnologija zrakoplova i senzora. Letjelice poput *PathFinder*-a služe za zadatke istraživanja atmosfere, kao i za komunikacijsku platformu. *PathFinder*-i su kasnije razvijeni u nove NASA Centurion i NASA Helios zrakoplove.

Slika 5: Bespilotna letjelica PathFinder [14]

Osim uloga poput poljoprivredne kontrole, pomoći u prirodnim katastrofama i humanitarnoj pomoći, bespilotne letjelice (UAV) se koriste i u državne, tj. vojne svrhe obavještavanja, nadzora, stjecanja ciljeva i izvidnice. Takvi dronovi koji uz dodatnu opremu nose zrakoplovna ofenzivna sredstva kao što su projektili ili bombe se nazivaju Bespilotne borbene letjelice (*eng. Unmanned combat aerial vehicle, UCAV*). Za razliku od bespilotnih borbenih letjelica, vrsta letjelice koja se koristi isključivo za obavještavanje, nadzor i izvidnicu, bez dodatnih ofenzivnih sredstava su bespilotne letjelice za nadzor i izviđanje. Primjer takve letjelice su poznate ISTAR letjelice koje se koriste u svrhu suzbijanja zločina, graničnih patrola, obalne straže, mapiranja i sličnih poslova koji zahtijevaju dodatnu sigurnost i pouzdanje u bespilotnu letjelicu i njene funkcionalnosti. Vrsta letjelica, poput bespilotnih borbenih letjelica i bespilotnih letjelica za nadzor i izviđanje, se postavljaju u posebnu domenu bespilotnih letjelica koja se zove MALE UAV *(medium-altitude long-endurace UAV)*. Domena MALE bespilotnih letjelica lete u visini između 3,000 do 9,000 metara s kapacitetom leta od 24 do 48 sati. Veliki broj MALE bespilotnih letjelica koriste Ploču daljinski upravljanih zrakoplovnih sustava (RPASP). Iako se još uvijek definiraju kao bespilotne letjelice budući da je upravljač van letjelice, ove bespilotne letjelice još uvijek zahtijevaju imati implementirane funkcionalnosti poput kontrolne veze (C2Link) i sustava za otkrivanje i izbjegavanje (DAA). Isti sustavi su implementirani u običnom zrakoplovu sa pilotom.

Slika 6: Bespilotna letjelica Istar [15]

Slika 7: Bespilotna letjelica Ucav [16]

Bespilotne letjelice koje lete na vrlo niskoj razini te nemaju potrebu za osiguranje zračnog prostora se nazivaju Small UAS. Sama ideja malih bespilotnih letjelica nastala je 1990-ih kada se raspravljalo o mogućim *'mobilnim mikrorobotima'* koje bi se koristile u vojnim operacijama. Ideju su u početku primili s puno skepticizma, ali se kasnije ispostavila kao tehnološka revolucija u vojnoj strategiji. Većina ovakvih letjelica se razvijaju pod pokroviteljstvom DARPA agencija (*eng. Defense Advanced Research Projects Agency*). Neki od rezultata njihovih projekta uključuju bespilotne letjelice poput *NANO AIR VEHICLE* i *HawkEye*. *NANO AIR VEHICLE* je dio projekta u kojem se razvija vrsta bespilotnih letjelica sposobne za obavljanje operacija van i unutar prostorija, koristeći mimiku ptice. Glavna karakteristika ove bespilotne letjelice je da je vrlo mala, te bi se koristila u svrhe izviđanja.

Slika 8: Bespilotne letjelice Nano Air Vehicle(lijevo) i HawkEye(desno) [17]

S druge strane, *HawkEye* je dizajniran za 'tajnu isporuku kritičnog tereta' zemaljskom osoblju korištenjem zračnog lansiranja s velike visine. Iako svrstana u *Small* UAS, ova bespilotna letjelica može potpuno autonomno isporučiti od 80 km udaljenosti. *HawkEye*. Podrazumijeva se da se ovakve bespilotne letjelice kontroliraju upravljačem, te su s time dio RPV bespilotnih letjelica.

3.1. Dronovi

Kao što je bilo spomenuto, definirati naziv za pojedine bespilotne letjelice je u nekim slučajevima nejasno. Mogli bi se reći, na najapstraktnijoj razini definicije, da je to letjelica bez pilota te ima ugrađene određene senzore radi sigurnosti i upravljanja. S druge strane, mnogi koriste pojam Dron kako bi opisali širok raspon vozila (*eng. Vehicles*) koja se kreću po zemlji, moru i u zraku. Kada je bilo spomenuto kako su sve bespilotne letjelice dronovi, ali svi dronovi nisu bespilotne letjelice [2], mislilo se na domenu dronova koji se ne kreću po zraku. S obzirom da se u ovom radu koristi ova vrsta bespilotnih letjelica, iskoristit ćemo definiciju koja govori kako je dron vrsta bespilotne letjelice koja ima implementiranu putanju te svojim funkcionalnostima može pratiti putanju ili, ukoliko je to potrebno, vratiti se na početnu poziciju bez ljudske intervencije. Dronovi se danas najčešće koriste za 3D mapiranje, fotografiranje i slične zadatke.

3.2. Swarm

Ideja jata dronova prvenstveno je proizašla iz ciljeva vojnih operacija. Prva poznatija takva operacija dogodila se 2021. godine kada su Izraelske Obrambene snage koristile jato dronova za vrijeme konflikta sa Hamas-om, palestinskom sunitsko-islamističkom fundamentalističkom organizacijom [7]. Naime, nasuprot popularnom vjerovanju da je jato dronova samo „veliki broj dronova koji se koristi odjednom“, prava definicija opisuje jato kao kompleksni sistem međusobne komunikacije i kolaboracije u donošenju kolektivnih odluka pri letu. Ovakva organizirana skupina dronova se naziva jato ili roj dronova.

> *„A system is more than the sum of its parts; it is an indivisible whole.“ — Russell L. Ackoff [6]*

Drugim riječima, jato dronova bi se moglo kategorizirati kao skupina dronova, koja funkcionira kao cjelina, gdje svaki dron ima svoju ulogu prema jatu. Prema dokumentu Američkog Ratnog Zrakoplovstva, 'rojenje' se definira *kao „Skupina autonomnih umreženih SUAS-a [sustava malih bespilotnih letjelica] koji rade u suradnji radi postizanja zajedničkih ciljeva s operaterom koji je uključen ili je u krugu“ [8].* Kada bi se spominjao točan broj dronova koji čini roj, količina je relativna; s obzirom na cilj i svrhu roja. Ono što je ključno je već spomenuta međusobna komunikacija, tj. međusobno dijeljenje informacija sa pojedinačnih senzora bespilotnih letjelica koji su članovi roja i donošenje kolektivnih odluka vođenih pomoću umjetne inteligencije.

3.2.1. Svrha

Nije upitno da je potražnja za razvojem rojeva dronova izrazito velika od strane vodećih visoko razvijenih zemalja koje ciljaju na korištenje rojeva u vojne svrhe, budući da ovakvi rojevi prognoziraju revolucioniranje ratovanja. Spomenuto je kako se govori isključivo o razvijenijim zemljama, tj. onima sa visokotehnološko razvijenim vojskama, s obzirom da stvaranje funkcionalnog i efikasnog roja bespilotnih letjelica zahtijeva vrhunsku tehnologiju softvera i hardvera [9]. Može se zaključiti kako je ključ izradbe dobrog 'uma košnice' među dronovima moćna umjetna inteligencija, napredni senzori i efikasne i brze podatkovne veze među bespilotnim letjelicama. Rojevi bespilotnih letjelica i njihovu usklađenost potražuju i unutar domene zabave i organiziranih događaja. Jedan takav primjer su svjetlosne predstave dronova koje izvode osvijetljene, sinkronizirane i koreografirane skupine dronova koje se slažu u različite zračne formacije. Za ovakav roj nije potrebna vrhunska umjetna inteligencija koje se traži na razini prijašnje spomenutih vojnih operacija.

Ovakvi događaji i predstave 'sinkroniziranog leta' se temelje na računalnom programu koji pretvara grafiku u naredbe za let te ih onda prenosi dronovima.

3.2.2. Formacije i komunikacija

Kod dizajniranja rojeva bespilotnih letjelica, najviše se zahtijeva pouzdani sustav upravljanja roja. U situacijama potrage i spašavanja, koordinacije ili određenih (vojnih) operacija, kontrola roja mora biti efikasna i precizna. Na prvi pogled, najjednostavniji način za sveukupno kontroliranje roja je korištenje jednog središnjeg računala ili pilota na tlu. Takva metoda se u praksi najčešće koristi u već spomenutim svjetlosnim predstavama bespilotnih letjelica. Postoje takvi pristupi koji se temelje na tehnici navigacije s jednim dronom koji, u svrhu izbjegavanja prepreka, preslikava rute koje dolaze unutar njegovog određenog radijusa. Preslikana ruta je zatim dostupna za korištenje ostalim dronovima koji su članovi roja koji mogu odabrati preslikanu i provjerenu rutu te na taj način izbjeći sudare dok ostaju u formaciji.

Iako je ovakav pristup dovoljno efikasan za spomenute potrebe, postoje uloge koje bi rojevi obavljali gdje se preferira autonomnija kontrola; poput praćenja kvalitete zraka. Drugi problem koji može izrasti iz ove metode je vrijeme koje je potrebno za izračunavanje i vršenje određenih prilagodbi kako ne bi došlo do kolizija među bespilotnim letjelicama. Vrijeme koje je za ulogu koju izvršava roj dragocjeno i čiji procesni izračuni usporavaju cijeli roj. Nova ideja koja se temelji na sinkroniziranom ponašanju rojeva iz prirode; poput jata ptica ili rojeva pčela. Ove životinje se kreću bez središnjeg računala ili centralne točke koja prikuplja podatke pa time kontrolira kretanja bespilotnih letjelica. Isto tako niti jedna pčela ne usmjerava kretanje za ostale ili senzorima odmiče druge pčele kako ne bi došlo do kolizije. Radije, one same izračunavaju vlastitu putanju kretanja na temelju leta svojih susjednih članova roja. Takve sinkronizacije se provode pomoću takozvanog Prediktivnog Izračunavanja ili Prediktivna Kontrola. Sam algoritam od inženjera matematike i doktoranda robotike sa švicarskog Federalnog instituta za tehnologiju u Lausannei (kratica. EPFL) Enrica Soria. Njegov algoritam osigurava bolje planiranje bez dodatnog usporavanja roja kao cjeline. Dronovi međusobno komuniciraju i interpretiraju pokrete u stvarnom vremenu kako bi se prognozirali i predvidjeli kamo se kreću drugi članovi roja. Ova metoda se dokazala efikasnijom od „reaktivnih“ kontrola za dronove bez prediktivnog izračunavanja, točnije 57% brža od kontrola.

"Ako želite u potpunosti implementirati ove stvari, trebali bismo

stvarno smanjiti potrebu za komunikacijom sa središnjim čvorištem

ili računalom", kaže Amir Barati Farimani, profesor strojarstva na

Carnegie Mellonu. [10]

4. Tehničke upute

Tehničke upute u nastavku opisuju način tehničkog oblikovanja rada, detaljnijeg opisa sustava za definiranje konstelacija bespilotnih letjelica i načina na koji sustav funkcionira. Tokom razvoja programa primarno se koristio alat Unity te za vizualizaciju dijagrama Visual Paradigm. U svrhu detaljnije obrade pojedinih skripti koje su se koristile u Unity, načine njihovog rada i funkcioniranja ćemo opisati koristeći već poznati IEEE Std 830-1998 standard koji spominje SRS (*eng. Software Requirements Specifications*). Ovim modelom zapravo specificiramo softverske zahtjeve, a kao krajnji rezultat se dobiva dokument s točno određenim i svima razumljivim specifikacijama za sustav koji se gradi. Iako se SRS najčešće koristi za dodatnu pomoć pri specificiranju određenih želja i potreba vezane za server sa strane kupca ili korisnika, u ovom radu se koristi kao *'outline'* za sustav i jednostavniji pregled dijelova sustava i način na koji on radi.

4.1. Unity alat

Unity je najviše poznat kao cross-platform *(hrv. više-platformski*) softver razvijen od strane Unity Technologies-a za izradu video-igrica i simulacija. Trenutno je dostupan na 27 platformi, zbog svojih drag-and-drop funkcionalnosti, besplatnog korištenja i skriptiranja u C# jeziku trenutno među najpopularnijim softverima unutar svoje domene. Zbog spomenutih karakteristika se ovaj softver koristio za razvoj sustava za definiranje konstelacija bespilotnih letjelica.

Slika 9: Unity Engine: Projekt Sustava za Definiranje Konstelacija Bespilotnih Letjelica, Screenshot: 26.8.2022.

4.2. Okolina sustava

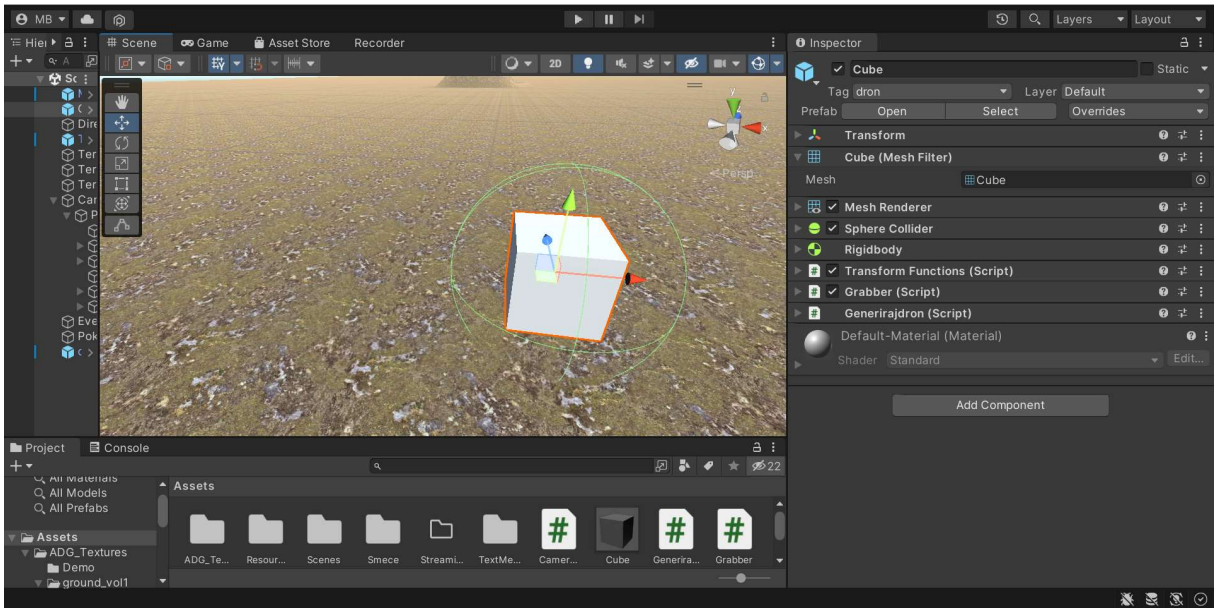
Projekt sadrži četiri scenarija:

Scene1	Scenarij koji služi kao uvod u aplikaciju; pozadina sa gumbom za početak definiranja konstelacija unutar simulacije.
Scene2	Scenarij unutar kojeg se postavlja n-broj bespilotnih letjelica (objekt „Cube“) te se postavljaju u početnu konstelaciju, nakon obavljanja početne konstelacije isti dronovi se postavljaju u završnu konstelaciju te se pokreće simulacija pomicanja bespilotnih letjelica, tj. n dronova.
Scene3	Scenarij u kojem se prikazuje pomicanje dronova. Unutar ovog scenarija koriste se okidači (eng. Triggers) ukoliko dolazi do potencijalnih kolizija.

Tablica 1: popis scenarija simulacije

Scenariji se koriste kako bi lakše pratili faze sustava. Isto tako olakšava korištenje Triggera i funkcija za određeni scenarij. Scene2 i Scene3 sadrže najviše funkcionalnosti i zajedno čine glavni dio sustava. Unutar njih se postavlja određeni broj dronova i njihove početne koordinate. Nakon postavljanja završnih koordinata dronovi obavljaju svoju konstelaciju te se konstantno provjeravaju potencijalne kolizije. Ukoliko dođe to potencijalne kolizije, dron koji u svojem nazivu ima manji redni broj se zaustavlja, mijenja boju u plavu i propušta dron s višim rednim brojem da prođe. Funkcionalnosti čekanja će se detaljnije obraditi kasnije.

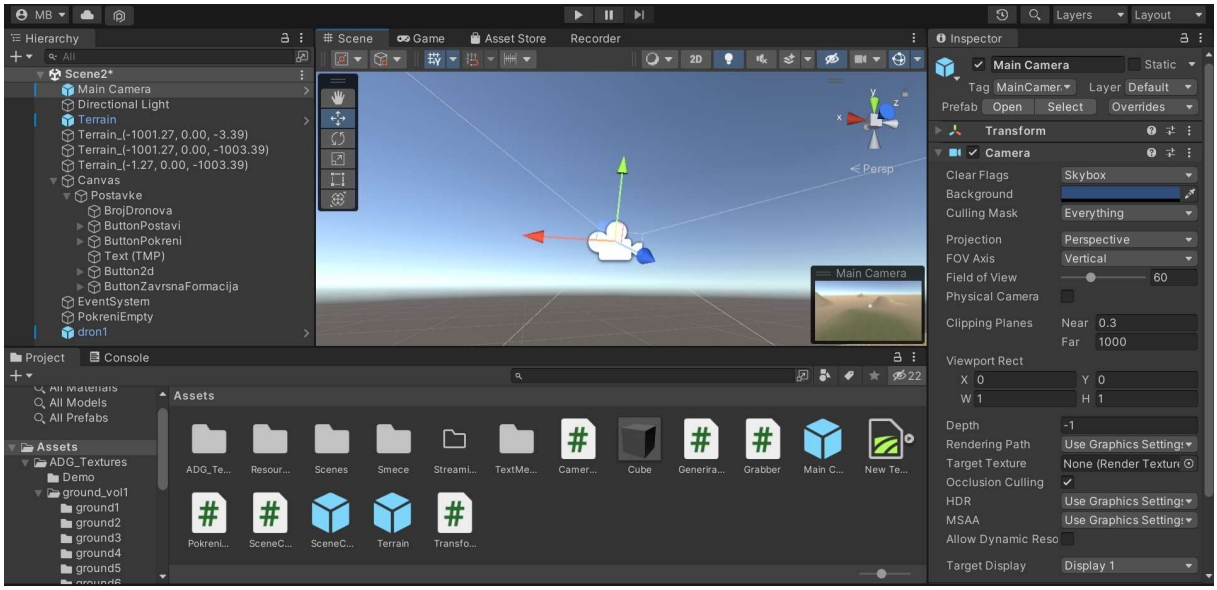
4.2.1. Entitet Cube



Slika 10: Entitet 'Cube' u Unity Softveru (ScreenShot 26.8.2022.)

Sustav koji je razvijen u Unity softveru bazira se oko jednog entiteta „Cube“ (prikazan na slici ___) koji se po potrebi klonira i identificira kao dron n+1. Entitet sadrži četiri komponente koje su obavezne kako bi instancirani dronovi koji koriste entitet „Cube“ pravilno funkcionirali u Scenarijima; Cube(Mesh Filter), MeshRenderer, Sphere Collider i Rigidbody. Od njih četiri najveću važnost ima Sphere Collider koji se koristi kao Trigger, tj. senzor tokom Scenarija 3 te provjerava je li dronov put 'čist' ili se dron treba zaustaviti kako nebi došlo do kolizije. Dodatno, entitet „Cube“ sadrži 3 skripte pomoću kojih obavlja svoje funkcionalnosti; Transform Functions, Grabber i Generirajdron, koji će se detaljno opisati kasnije.

4.2.2. Main Camera



Slika 11: Main Camera Object u Unity softveru, Screenshot 26.8.2022

Object (GameObject; tip objekta u Unity softveru*) Main Camera* koristi se kao korisničko sučelje, tj. korisnikov pregled na okolinu. Unutar njega koristi se skripta *Camera Focus. Main Camera* je prisutna u svim scenarijima te ima dva dostupna stanja u kojima može biti.

4.2.3. PokreniEmpty (EmptyGameObject)

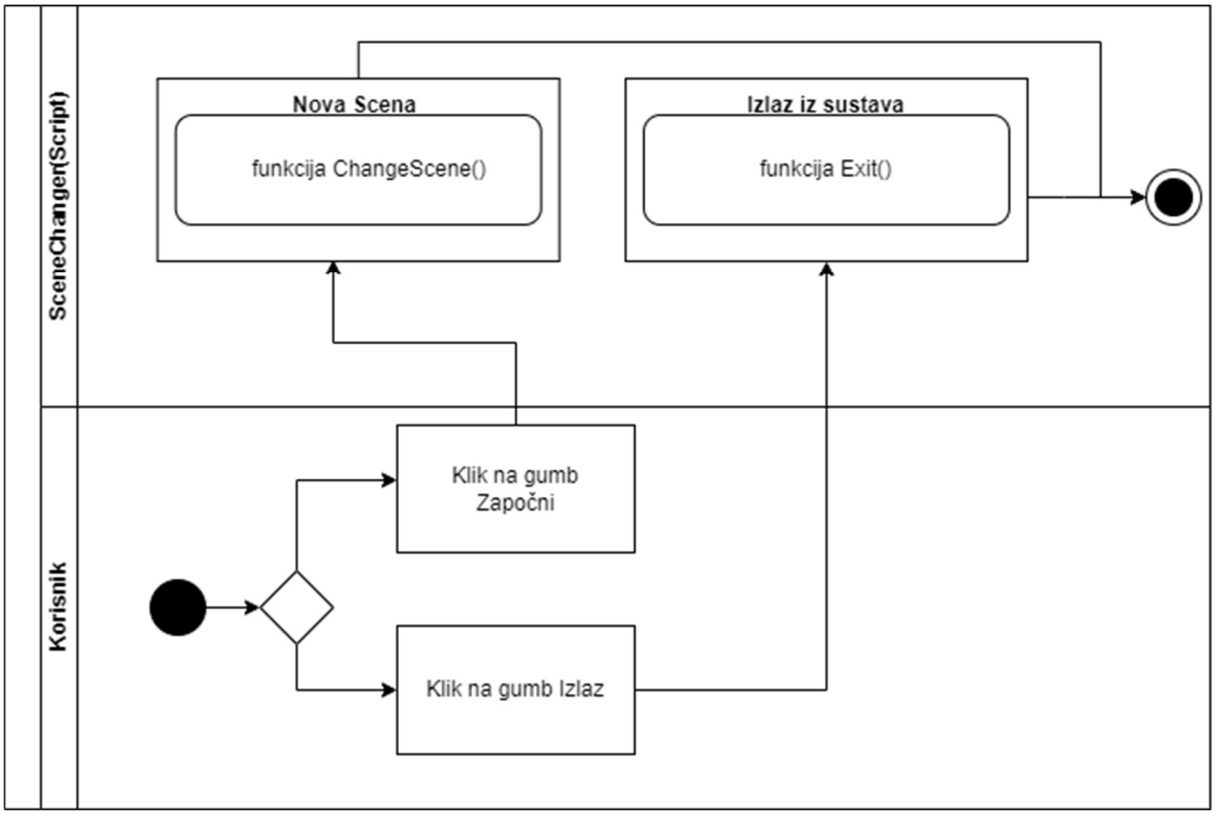
PokreniEmpty je nevidljivi objekt koji se koristi primarno zbog skripte *PokreniScript* koja se uvijek nalazi unutar *PokreniEmpty* objekta. Iako nevidljiv, *PokreniEmpty* je zbog svoje skripte krucijalan za rad cijelog sustava. *PokreniScript* provjerava početne i završne formacije postavljenih dronova i kalkulira njihovo pomicanje i izbjegavanje kolizija.

4.3. Funkcionalnosti sustava

Funkcionalnosti sustava opisuju specifične funkcionalnosti koje se koriste i procese unutar funkcionalnosti koji se provode po zahtjevu korisnika. Uz opis funkcionalnosti se prikazuju i dijagrami koji vizualno detaljnije prikazuju procese.

4.3.1. Funkcionalnosti Započni i Izlaz

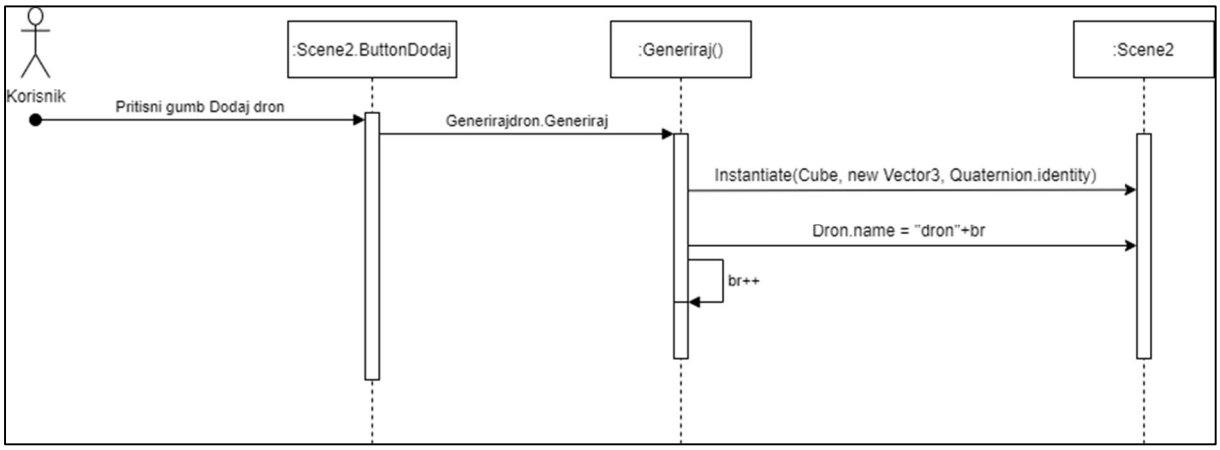
Funkcionalnosti gumba započni i gumba izlaz nalaze se u prvoj sceni sustava (Scene1). Dijagram prikazuje obje funkcionalnosti budući da im se može pristupiti na jednakoj razini sustava. Funkcionalnosti Započni i Izlaz su jedina dva izbora koja se nude neposredno nakon pokretanja samog sustava.



Dijagram 1. Funkcionalnosti Započni i Izlaz (Izrađeno 27.8.2022.)

4.3.2. Funkcionalnost dodavanja dronova

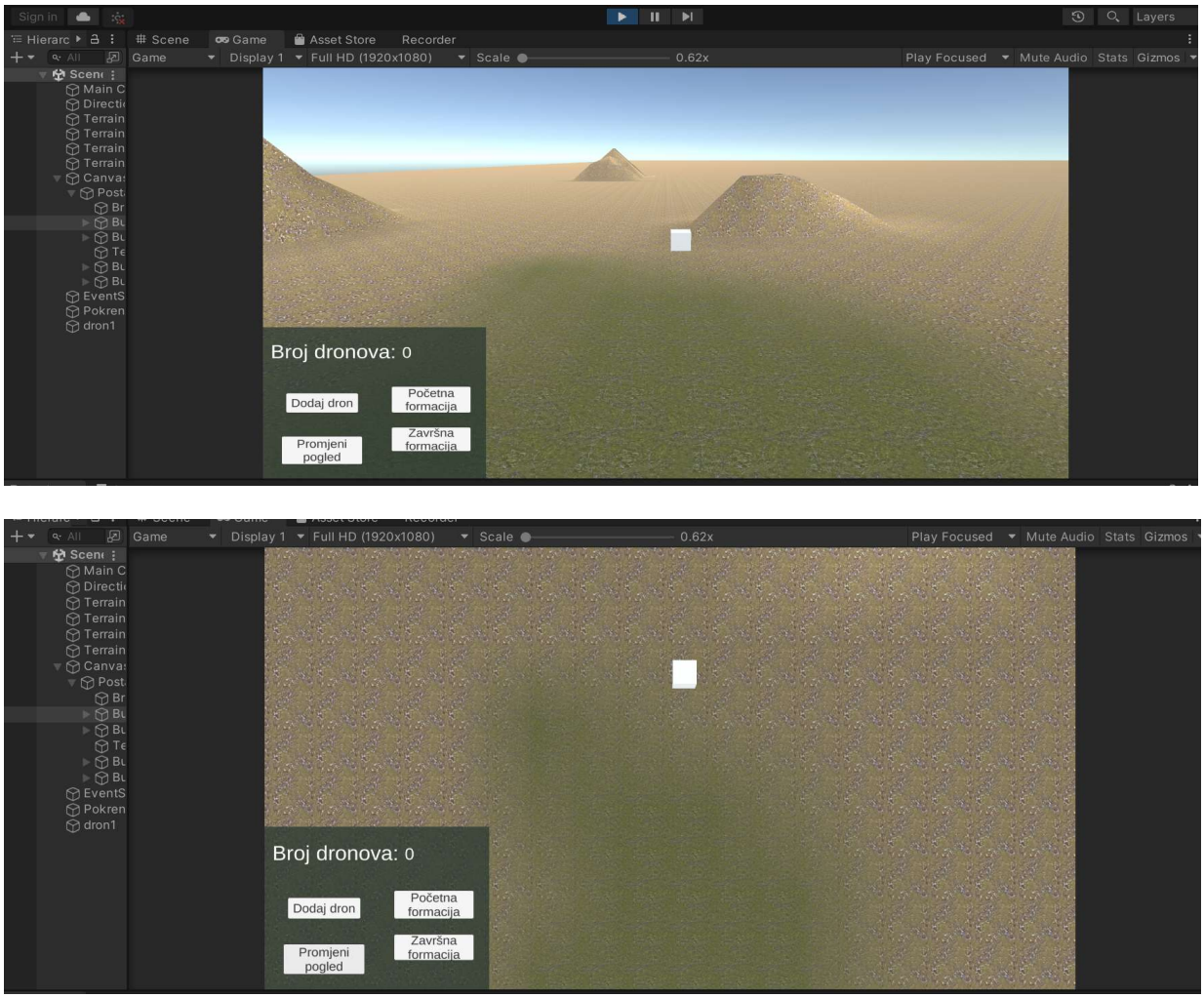
Funkcionalnost dodavanja dronova obavlja se pritiskom na gumb „Dodaj dron“ unutar druge scene sustava (Scene2). Prilikom pritiska, gumb šalje zahtjev i time pozive funkciju Generiraj() unutar skripte Generirajdron. Funkcija Generiraj poziva funkciju Instantitate; funkcija koja je već ugrađena u Unity softveru te kojom se generira nova instanca. U ovom slučaju generiramo instancu Cube (entitet koji se spominjao prije) sa novim položajem koji opisuje drugi argument funkcije Instantiate (Vector3 je položaj koji u sebi ima tri argumenta, tj. odabir koordinata). Funkcija Generiraj isto tako pridodaje novo-generiranoj instanci Dron novo ime; „dron“+br. U skripti se spominje instansa br koja služi kao brojač te počinje s nulom (int br = 0). Razlog tome je što na kraju svakog poziva funkcije Generiraj() instanca br se podiže za jedan, te se time svaki novi dron naziva „dron“ sa dodatnim novim rednim brojem. Ova tehnika će kasnije dobro poslužiti za biranje koji dron se mora zaustaviti i propustiti drugi tokom same simulacije leta.



Dijagram 2.: Funkcionalnost dodavanja dronova (Izrađeno 27.8.2022.)

4.3.3. Funkcionalnost promjene pogleda

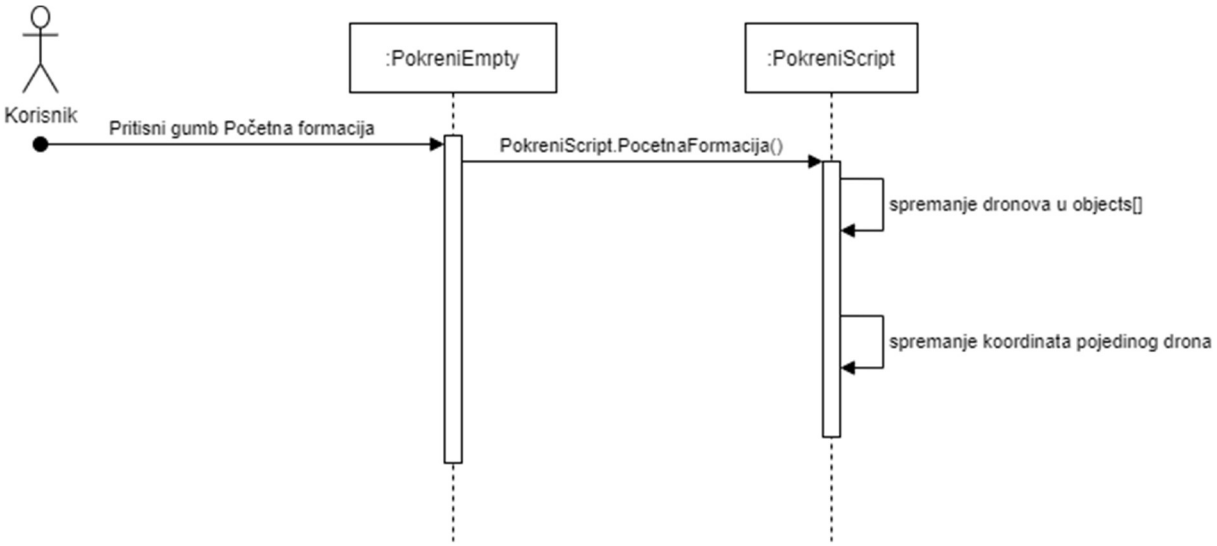
Funkcionalnost promjene pogleda odnosi se na objekt kamere koju smo prije spominjali. Funkcionalnost nudi dva stanja pogleda kamere sa unaprijed definiranim koordinatama. Razlog tome je što početna okolina može biti nepregledna. Isto tako, za sustav je lakše koristiti donju perspektivu radi postavljanja dronova na njihovu početnu i završnu formaciju, tj. konstelaciju. Budući da korisnik simulacije koristi miš za pomicanje objekata, tj. dronova, precizniji način je pozicioniranje dronova iz perspektive slike 12.



Slika 12 i Slika 12: dva moguća prikaza kamere tokom simulacije, ScreenShot 27.8.2022.

4.3.4. Funkcionalnost početne formacije

Funkcionalnost početne formacije te poziva pritiskom na gumb Početna formacija unutar druge scene sustava (Scene2). Pritiskom na gumb već spomenuti prazni objekt (eng. empty GameObject) poziva funkciju PocetnaFormacija() unutar skripte PokreniScript. Ta funkcija obavlja 2 aktivnosti; spremanje dronova u polje objects i spremanje koordinata pojedinog drona. U skripti PokreniScript postoji već ugrađena funkcija Update() koja se konstantno provjerava i ažurira dok je sustav aktivan i radi (eng. per frame). Unutar nje je deklarirana instanca niza objects koji unutar sebe stavlja sve instancirane dronove. Na taj način niz objects će vrlo precizno provjeravati ima li u sebi sve aktivne dronove koji se nalaze u simulaciji. Osim u funkciji Update(), provjeravanje točnosti niza objects se provodi na više mjesta unutar skripte PokreniScript radi dodatnog opreza.

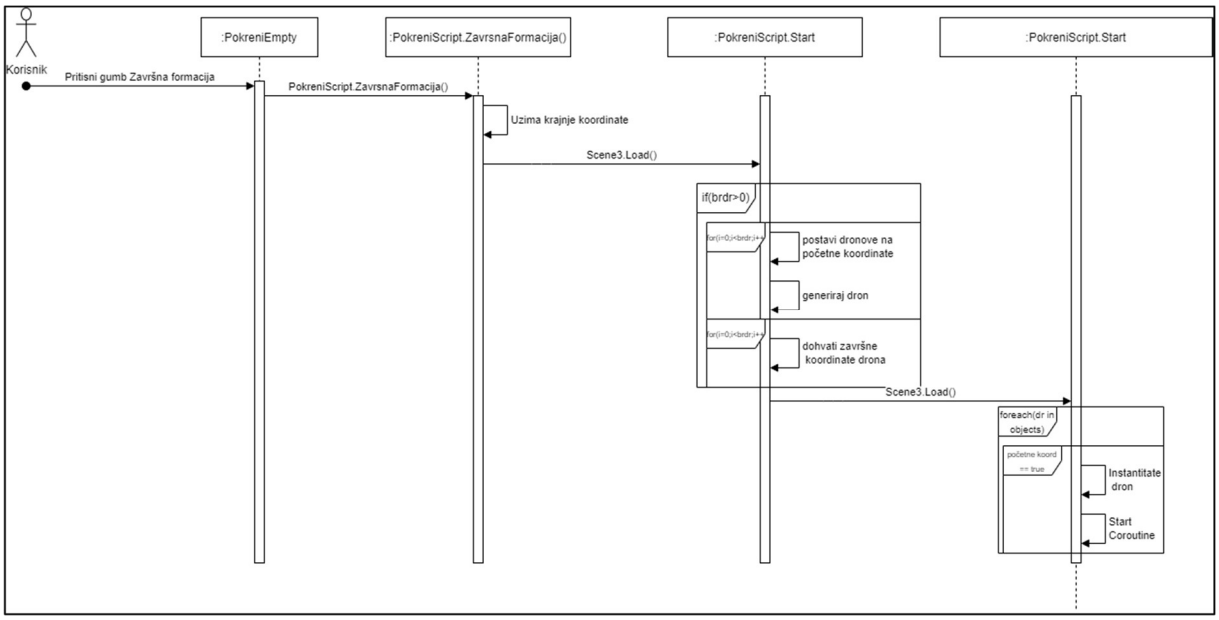


Dijagram 3.: Funkcionalnost postavljanja početne formacije (Screenshot 27.8.2022.)

4.3.5. Funkcionalnost završne formacije

Funkcionalnost završne formacije opisuje procese koji slijede nakon pritiska gumba završne formacije. Svi procesi se provode unutar skripte PokreniScript. Najprije se unutar funkcije ZavrsnaFormacija() postavljaju krajnje koordinate svakog drona, tj. očitavaju se koordinate na kojima se nalaze dronovi iz druge scene (Scene2). Nakon toga se otvara prikaz treće scene (Scene3) te se provjerava instanca brdr (skraćeno broj dronova), tj. broj dronova. Ukoliko je veći od nula može se nastaviti sa pripremom dronova i njihovih smještaja za simulaciju. Za svaki dron koji se nalazi u nizu objects koji je jednak instanci brdr isti se postavlja na početnu poziciju koja je utvrđena preko prijašnje funkcionalnosti za početne formacije. Neposredno nakon toga se ponovo generiraju dronovi unutar treće scene te se isto tako spremaju završne koordinate, tj. koordinate na koje se dron mora preko simulacije premjestiti.

Kada dohvatimo pojedini dron koji ima definirane početne i krajnje koordinate i brzinu, koristimo funkciju koja je već ugrađena u Unity softver StartCoroutine(MoveObject(dron, krajnjaTocka, brzina);.



Dijagram 4.: funkcionalnost završne formacije (Screenshot 27.8.2022.)

5. Algoritam prevencije kolizija

5.1. Funkcija Coroutine

Kao što je već spomenuto, glavna problematika ideje roja bespilotnih letjelica, kao i glavno rješenje sustava za definiranje konstelacija bespilotnih letjelica, je izbjegavanje kolizija dronova unutar roja. U ovome radu su spomenute vrste sinkronizacija i međusobnog izbjegavanja pomoću Prediktivnog Izračunavanja ili Prediktivna Kontrola. Prediktivnom kontrolom dronovi izbjegavaju koliziju time da su dodatno 'svjesni', tj. sposobni izračunati kretanje drugih članova roja. Unatoč dokazanoj efikasnosti ove metode, za potrebe simulacije koristila se pojednostavljena metoda izbjegavanja kolizija bespilotnih letjelica. Jedna od ključnih funkcija, koja je među već integriranim i unaprijed definiranim funkcijama unutar Unity softvera, se naziva *StartCoroutine().* Općenito, Coroutine (hrv. Korutina) je vrsta funkcije koja omogućuje trenutno zaustavljanje nekog objekta unutar simulacije ili izvođenje u određenom trenutku te njen nastavak nakon ispunjenog uvjeta. To je jedan od načina zaustavljanja izvršenja neke aktivnosti dok se ne ispuni predefinirani uvjet nakon kojega nastavlja od mjesta gdje je stala. Dok kod većine funkcija se može postaviti vraćanje vrijednosti bilo kojeg tipa, kod funkcija Coroutine se zahtjeva vraćanje vrijednosti IEnumerator. Sukladno tome, unutar funkcije Coroutine, umjesto 'return' se koristi naredba 'yield'.

```
IEnumerator Korutina()
{
  Debug.Log("Hello world");
  yield return null;
}
```

Ukoliko je unutar funkcije definirana naredba yield return null, podrazumijeva se da se aktivnosti nastavljaju nakon što se sve unutar Update() funkcije izvrši. Update funkcija će se detaljnije obraditi kasnije u radu. Dodatno tome, tip vrijednosti zvan IEnumerator je vrsta klase koja u ovome slučaju služi kao petlja poput *for* ili *foreach.* IEnumerator vraća true ili false. Unutar Unity softvera, Coroutine i IEnumerator mogu pravilno raditi ukoliko su priključeni na MonoBehaviour kao svoj nadređeni proces. On provjerava uvjete spomenutih funkcija kroz svaki frame (hrv. okvir) te se time prate uvjeti unutar simulacije. MonoBehaviour je za Unity softver temeljna klasa koja se koristi unutar programa. U slučaju sustava za definiranje konstelacija bespilotnih letjelica, MonoBehaviour se koristi za svaku skriptu unutar simulacije. Najčešće funkcije koje su podređene klasi MonoBehaviour a pronalaze se u simulaciji su

funkcije poput Start(), Update() i Awake(). Jedna od glavnih karakteristika ovih funkcija je da su aktivne u svakom frame-u trajanja simulacije.

5.2. Algoritam Sustava

Sustav za definiranje konstelacije bespilotnih letjelica koristi funkciju StartCoroutine kao jednu od glavnih načina prevencije kolizija bespilotnih letjelica unutar simulacije. Također, funkcija StartCoroutine() poziva dodatnu funkciju koja je također ugrađena u Unity softver; MoveObject;

```
IEnumerator MoveObject(GameObject dr, Vector3 target, float overTime){
        float startTime = Time.time;
        while(Time.time < startTime + overTime){
            if(speed == overTime){
                dr.transform.position                                    =
Vector3.MoveTowards(dr.transform.position,     target,     (Time.time     -
startTime)/overTime);
                bool staoo = dr.GetComponent<TransformFunctions>().stao;
                Debug.Log(staoo);
                if(staoo == true ){
                    yield return new WaitForSeconds(1);
                    dr.GetComponent<TransformFunctions>().stao = false;
                }
                else if(staoo == false){
                    yield return null;
                }
            }
            else if(speed != overTime){
                overTime = speed;
            }
        }
        dr.transform.position = target;
    }
```

U prikazanom kodu vidi se instanca tipa 'bool' pod nazivom *stao* koja se preuzima iz skripte TransformFunctions koja je povezana na entitet Cube pomoću kojeg se generiraju dronovi. Instanca bool je krucijalna za pravilan rad prije spomenutih senzora koje koristimo kako bi pratili da ne dolazi do kolizija. Ukoliko dron prepoznaje susjedni dron koji ima veći redni broj te ukoliko taj susjedni dron dolazi u doticaj sa dronovim već prije spomenutim

Colliderom koji se koristi samo kao Trigger za senzor, tada spomenuti dron mijenja boju u plavu i prestaje se kretati na jednu sekundu. Unutar prijašnjeg primjera koristi se naredba yield return new WaitForSeconds(1). Ovime se osigurava da objekt ostane na mjestu jednu sekundu, nakon čega pomoću naredbe koja slijedi izlazi is stanja *'stao'* te se nastavlja kretati kroz svoju putanju.

Stanje *stao* je definirano unutar skripte TransformFunctions(). To je skripta koja se postavlja unutar svakog drona koji se kreira unutar simulacije te je postavljena kao *false*. Dio programa koji se koristi u funkciji TransformFunctions za pravilno funkcioniranje senzora koji :

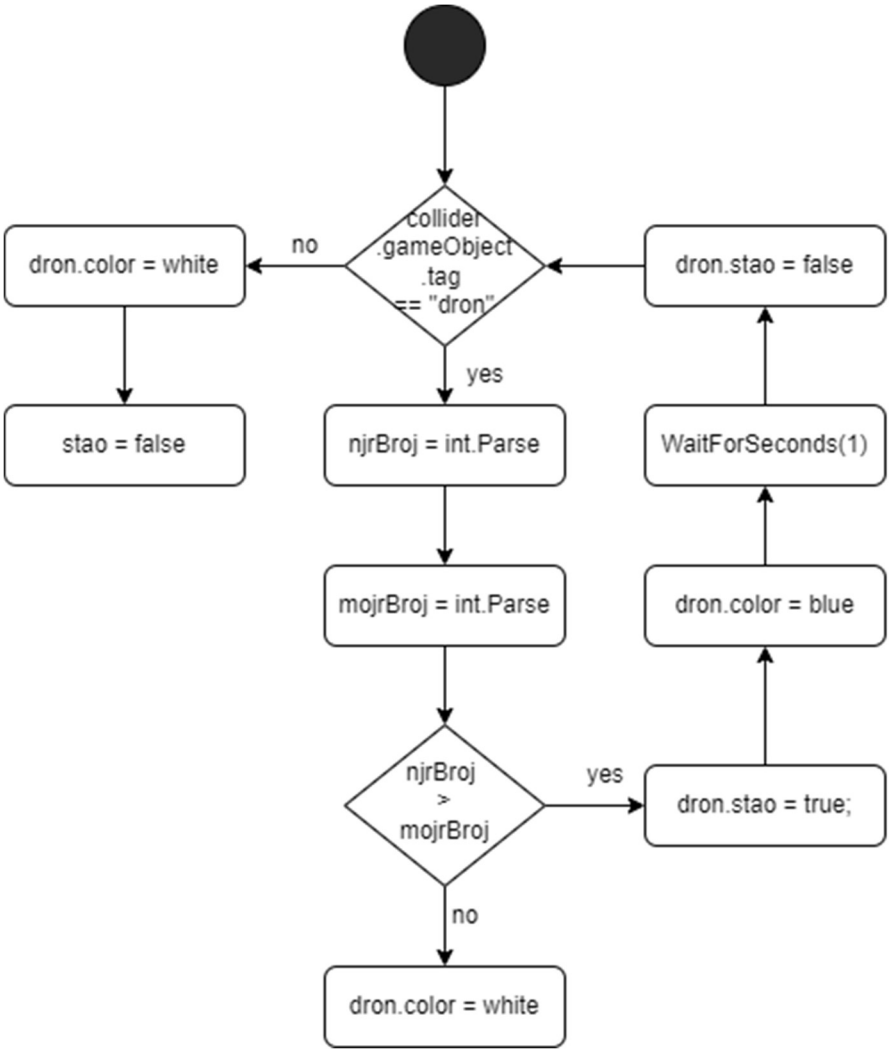
```
void Update(){
        foreach(Collider col in currentrlyCollidingWith){
            if(col.gameObject.tag == "dron"){
                float             njrBroj             =             (float)
int.Parse(col.gameObject.name.Substring(4));
                float            mojrBroj             =             (float)
int.Parse(gameObject.name.Substring(4));

                if(njrBroj > mojrBroj){
                    stao = true;
                    GetComponent<Renderer>().material.color = Color.blue;
                }
            } else{
                GetComponent<Renderer>().material.color = Color.white;
                stao = false;
            }
        }
    }
```

TransformFunction kao skripta koja je većinski odgovorna za pravilno ponašanje senzora pojedinog drona isto tako ima dodatne dve funkcije koje se koriste za pravilan rad senzora. Funkcije OnTriggerEnter() i OnTriggerExit(). Ove funkcije su, poput funkcija Update() i Awake(), isto tako već unaprijed ugrađene u Unity softveru. Funkcija OnTriggerEnter() reagira samo ukoliko dolazi do kolizije objekta koji sadrži skriptu sa spomenutom funkcijom. Razlika između funkcije OnCollisionEnter() koja se najčešće koristi kao senzor jest ta da se OnCollisionEnter stavlja na objekte koji imaju fizičku masu te se unutar simulacije sudare i reagiraju tako da se odbiju jedan od drugoga. U ovome djelu algoritma važno je napomenuti da su novo kreirani dronovi unutar simulacije automatski dobili svoj redni broj. S obzirom da svaki objekt unutar simulacije koji obavlja funkcije drona ima istu skriptu TransformFunction(),

najefikasnije rješenje je bilo da se unutar simulacije dron zaustavi i pričeka prolazak onog drona koji ima veći redni broj uz svoje ime.

Slijedi dijagram aktivnosti koji opisuje način na koji dron provjerava je li došlo do kolizije te pod kojim uvjetima se on zaustavlja a pod kojim uvjetima nesmetano nastavlja prolaziti kroz svoju predodređenu putanju.



Dijagram aktivnosti 5.: izbjegavanje kolizije, Izradio autor 2022.

Ukoliko dođe do kolizije sa objektom koji ima predodređeni tag, tj. naljepnicu 'dron', skripta provjerava redni broj oba drona. To radi tako da pomoću naredbe Parse provjeri redni broj unutar imena oba objekata. Naime, već je spomenuto da je ovaj odjeljak skripte svakome objektu isti. Zbog toga svaki dron u koliziji će definirati instancu njrBroj (skraćeno, njegov redni Broj) kao redni broj tuđih dronova te mojrBroj (skraćeno, moj redni Broj) kao vlastiti redni broj. U globalnoj perspektivi, svaki dron unutar kolizije će imati različite vrijednosti ovih dviju instanci u usporedbi s drugim dronovima. Ovo naime ne stvara problem budući da je uvjet univerzalan;

ukoliko je unutar skripte određenog drona instanca njrBroj veći, dron mora stati na jednu sekundu (WaitForSeconds(1)). Ukoliko ipak njrBroj nije veći, dron se ne zaustavlja. Zbog univerzalnog pravila koji određuje koji dron propušta drugoga (dronovi s manjim rednim brojem propuštaju dronove s većim rednim brojem), različite vrijednosti instanci nisu prepreka algoritmu. Na dijagramu se isto tako može vidjeti kako će dron koji se zaustavlja na trenutak poprimiti plavu boju radi lakšeg snalaženja korisnika i njegovog razumijevanja simulacije.

Odjeljak algoritma koji je dio skripte PokreniScript() te koji instancira novi dron sa novim rednim brojem slijedi:

```
GameObject Dron = Instantiate(Cube, new Vector3(x[i],y[i],z[i]),
Quaternion.identity);

                    Dron.name = "dron"+i;

                    TransformFunctions trans =
Dron.GetComponent<TransformFunctions>();

                    GameObject pokreniEmpty = GameObject.Find
("PokreniEmpty");

                    trans.pokreni = pokreniEmpty;
```

Unutar funkcije TransformFunctions() koriste se još neke krucijalne funkcije. Koristeći funkciju OnTriggerEnter() uz prethodno označavanje spomenutog entiteta Cube koji se koristi za instanciranje pojedinog drona kao Trigger, izbjegavamo fizičke reakcije dronova pri koliziji već senzori prije vidljive kolizije obavještavaju dron o stanci.

```
void OnTriggerEnter(Collider collInfo){

            if(collInfo.gameObject.CompareTag("dron")){
                Debug.Log("alo bre seljacino");
                pokreni.GetComponent<PokreniScript>().speed      =      200;
currentrlyCollidingWith.Add(collInfo.GetComponent<Collider>());
                lastCollidedObject = collInfo.gameObject;
            }
    }
    void OnTriggerExit(Collider collInfo){
        if(collInfo.gameObject.CompareTag("dron")){
            Debug.Log("sve pettt");
        }
        currentrlyCollidingWith.Remove(collInfo.GetComponent<Collider>());
    }
```

6. Zaključak

Predmet rada je definiranje bespilotne letjelice, roja te njihov detaljniji opis, vrste i svrha. Uz to se opisuje sustav za definiranje konstelacija bespilotnih letjelica. Glavna svrha sustava je izraditi simulaciju kretanja određenog broja bespilotnih letjelica. Kroz razne funkcije, neke koje su već unaprijed implementirane od strane korištenog Unity softvera dok su druge definirane od strane autora, sustav pokazuje početnu verziju simulacije.

Postavljanjem početne i završne formacije, tj. postavljanje pojedinog drona unutar simulacije na njegovo početno i završno mjesto koje korisnik proizvoljno bira, pojedini dron se kreće kroz svoju putanju to završne točke. Sustav se dijeli na tri scenarija, prvi koji služi kao uvod u simulaciju dok se druga dva scenarija koriste isključivo za pravilno funkcioniranje i rad same simulacije. Već spomenuta glavna problematika bila je izbjegavanje kolizije. Način na koji se kolizija izbjegava unutar sustava za definiranje konstelacija bespilotnih letjelica bio je da dron koji u svojem imenu ima manju brojčanu vrijednost propušta dron s većom brojčanom vrijednošću. Sam objekt koji se definira kao zaseban dron unutar simulacije je u obliku kugle, dok vizualno izgleda drugačije. Razlog tome je ušteda vremena tokom stajanja radi izbjegavanja kolizije. To je jedan od više primjera velikog broja isprobavanja i mjerenja načina na koji sustav funkcionira van domene čistih izračuna i točno određenih parametara.

Sustav za definiranje bespilotnih letjelica funkcionira na bazi čekanja i propuštanja dronova. Sam sustav je napravljen u 3D simulaciji ali se dronovi ne prelijeću, tj. putanje dronova i samo kretanje dronova je dvodimenzionalno. Kao što je već napomenuto, trenutna verzija projekta prikazuje samo osnovne funkcije kojima objekti, tj. bespilotne letjelice relativno izbjegavaju kolizije. Sustav još uvijek ima mjesta za nadogradnju i dodatni razvoj poput efikasnijeg izbjegavanja kolizija te mogućeg bržeg rezultata transformacija konstelacija.

Popis literature

[1] BBC, „Heathrow airport: Drone sighting halts departures“, 2019., [Na internetu] Dostupno na: https://www.bbc.com/news/uk-46803713 **[Pristupljeno 12.9.2022.]**

[2] Sam, „What's the Difference Between Drones and UAV's?“, 2019 [Na internetu] Dostupno na *What's the Difference Between Drones and UAVs? (dronesvilla.com)* [Pristupljeno 10.8.2022.]

[3] Ad Hoc Networks, „Medium Altitude Long Endurance“, 2021 [Na internetu] Dostupno na https://www.sciencedirect.com/topics/engineering/medium-altitude-long-endurance [Pristupljeno 10.8.2022]

[4] Filippo De Florio, „Airwothiness of Unmanned Aircraft Systems (UAS)“, 2016 [Na internetu] Dostupno na https://www.sciencedirect.com/topics/engineering/unmanned-aircraft-system [Pristupljeno 13.8.2022]

[5] Ad Hoc Networks, „High-Altitude Long Endurance“, 2021, Dostupno na https://www.sciencedirect.com/topics/engineering/high-altitude-long-endurance, [Pristupljeno 13.8.2022.]

[6] Russell L. Ackoff, „Science in the Systems Age: beyond IE, OR and MS.“, 1970., Operations Research Vol 21, str. 664.

[7] Zachary Kallenborn, „Swarm talk: understanding drone typology“, 2021 [Na internetu] Dostupno na https://mwi.usma.edu/swarm-talk-understanding-drone-typology/ [Pristupljeno 1.9.2022]

[8] UAF, „Small Unmanned Aircraft Systems (SUAS) Flight Plan: 2016-2036.“, 2016 [Na internetu] Dostupno na https://www.af.mil/Portals/1/documents/isr/Small_UAS_Flight_Plan_2016_to_2036.pdf [Pristupljeno 20.8.2022.]

[9] David Hambling, „What are drone swarms and why does every military suddenly want one?“, Forbes, 2021 [Na internetu] Dostupno na https://www.forbes.com/sites/davidhambling/2021/03/01/what-are-drone-swarms-and-why-does-everyone-suddenly-want-one/?sh=2eac7c5f2f5c [Pristupljeno 2.9.2022.]

[10] Max G. Levy, „Watch a Drone Swarm Fly Through a Fake Forest Without Crashing“, 2021 [Na internetu] Dostupno na https://www.wired.com/story/watch-a-drone-swarm-fly-through-a-fake-forest-without-crashing/ [Pristupljeno 5.9.2022.]

[11] Unity Logo [Slika]. Dostupno https://unity.com/ [Pristupljeno 12.9.2022]

[12] Bespilotna letjelica [Slika]. Dostupno https://express.24sata.hr/tehno/rusi-unistavaju-americke-letjelice-i-nitko-ne-zna-kako-15232 [Pristupljeno 12.9.2022.]

[13] Kettering bespilotna letjelica [Slika]. Dostupno https://www.nationalmuseum.af.mil/Visit/Museum-Exhibits/Fact-Sheets/Display/Article/198095/kettering-aerial-torpedo-bug/ [Pristupljeno 1.6.2022.]

[14] Dronovi i njihove uloge [Slika] Dostupno https://www.researchgate.net/figure/Potential-civilian-and-commercial-uses-for-small-UAS-11_fig1_342401654 [Pristupljeno 1.6.2022.]

[15] Bespilotna letjelica PathFinder [Slika] Dostupno https://www.avinc.com/innovative-solutions/hale-uas [Pristupljeno 1.6.2022.]

[16] Bespilotna letjelica Istar [Slika] Dostupno https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104470/mq-9-reaper/ [Pristupljeno 1.6.2022.]

[17] Bespilotna letjelica Ucav [Slika] Dostupno https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104470/mq-9-reaper/ [Pristupljeno 1.6.2022.]

[18] Nano Air i HawkEye [Slika] Dostupno https://www.microdrones.com/en/ [Pristupljeno 1.6.2022.]

Popis slika

Popis Tablica