Incze Attila, AZ5FVR attila.incze@gmail.com

1. A feladat kiírása

3D-ben mozgó, 6 szabadsági fokkal jellemezhető merev test szimulálása. A test egy speciális kialakítású modell-helikopter, a fellépő erők és forgatónyomatékok a motor/propeller által keltett húzóerő és forgatónyomaték, valamint a gravitációs erő.

A helikopter ún. quadrotor felépítményű, azaz négy motorral rendelkezik, melyek fordulatszáma nulla és egy maximum között szabályozható. A szimuláció lehetőséget ad egy szabályzó kód elhelyezésére, mely kódnak lehetősége van a motorok fordulatszámának változtatására. Bemeneti paraméterekként szenzorok adatait veheti alapul, melyeket a szimuláció biztosít számára. A rendelkezésre álló szenzorok: 3 tengelyű gyorsulásmérő szenzor, valamint 3 tengelyű szögsebesség szenzor.

2. Elméleti háttér áttekintése

A test orientációját a labor koordinátarendszerhez képest egy ortogonális 3x3 mátrix segítségével írhatjuk le: $\boldsymbol{A}(t)$.

Ennek pontosan 3 szabadsági foka van ezáltal alkalmas 3D elfordulás (szög) leírására.

Az orientáció mellett szükségünk van a test koordinátarendszeréhez rögzített derékszögű koordinátatengelyek mentén mért szögsebesség értékekre: $\boldsymbol{\omega}(t)$.

A test tömegközéppontjának mozgását egy $\boldsymbol{r}(t)$ helymátrixal, valamint $\boldsymbol{v}(t)$ sebesség mátrixal írhatjuk le. A testre ható erők eredője $\boldsymbol{F}(t)$, a forgatónyomaték pedig $\boldsymbol{\tau}(t)$. A test impulzusmomentuma $\boldsymbol{L}(t)$, a tehetetlenségi nyomaték tenzora pedig $\boldsymbol{I}(t)$.

A szimuláció lépései a kinematikai és dinamikai mozgásegyenletek diszkretizációjával az alábbi alakot öltik (h az időlépés):

$$\boldsymbol{r}(t+h) = \boldsymbol{r}(t) + h * \boldsymbol{v}(t)$$

$$\boldsymbol{v}(t+h) = \boldsymbol{v}(t) + h * \boldsymbol{F}(t)/M$$

$$\boldsymbol{A}(t+h) = \boldsymbol{A}(t) + h * \tilde{\boldsymbol{\omega}}(t) * \boldsymbol{A}(t)$$

$$\boldsymbol{L}(t+h) = \boldsymbol{L}(h) + h * \boldsymbol{\tau}(t)$$

$$\boldsymbol{I}(t+h) = \boldsymbol{A}(t+h) * \boldsymbol{I}^{-1} * \boldsymbol{A}^{\boldsymbol{T}}(t+h)$$

$$\boldsymbol{\omega}(t+h) = \boldsymbol{I}(t+h) * \boldsymbol{L}(t+h)$$

3. Számítógépes algoritmus kidolgozása / 4. A számítógépes kód leírása

Az algoritmus, mely a fent leírt képletekkel számol, nagyon egyszerű megvalósítású: A különböző dinamikai és kinematikai mennyiségeknek egy tömböt hozok létre, mely a különböző időpillanatokban tárolja az adott mennyiség értékét. Az időlépések száma választható: *ntime.* Természetesen szükség van még egy időlépés nagyságára: *h* (másodpercben).

Az algoritmus egy ciklus, mely *ntime* alkalommal fut le, és minden lépésben *h* időlépéssel végzi el a mennyiségek időfejlődését, valamint futtatja az autopilótát, mely korrekciót végez a rotorok meghajtásán.

A fellépő eredő erő és forgatónyomaték a négy rotor által keltett hatásból számolható. A négy rotor helyzete a szimuláció elején állítható, és mindegyik rendelkezik egy fordulatszámmal, mely nulla és egy maximum között változtatható. A fordulatszámmal lineárisan változó húzóerők adják az eredő erőt, és a fordulatszámmal lineárisan változó forgatónyomatékok adják az eredő forgatónyomatékot. A testnek súlya csak a négy motor helyén van, ez szintén egyenként állítható.

A szenzorok értékei alapján működik az autopilóta, azonba a szenzorok értékeit nekünk kell „elkészítenünk”. Mivel valós szenzorokat szeretnénk szimulálni, ezért aluláteresztő szűrőn átengedve, és zajjal keverve készíthetjük el a szenzorok jeleit a szimulációból kapott dinamikai értékekből.

5. A program használata

Az általam készített program, a *quadcopter_sim.m* fájl az alábbi részekre tagolódik: kezdőfeltételek megadása, szimuláció, grafikus megjelenítés. A program elején állíthatjuk tehát be a kezdőfeltételeket. A szimulációs rész elején definiálhatjuk a szabályzó algoritmust. A megfelelő paraméterek/szabályzás beállítása után futtathatjuk le a kódot, mely a kimenetet grafikusan mutatja. Különböző dinamikai és kinematikai mennyiségeket ábrázol az eltelt idő függvényében, majd egy animációval szemlélteti a helikopter időbeli mozgását. Ezzel szemléletesen is láthatjuk a stabilázálás hatékonyságát.

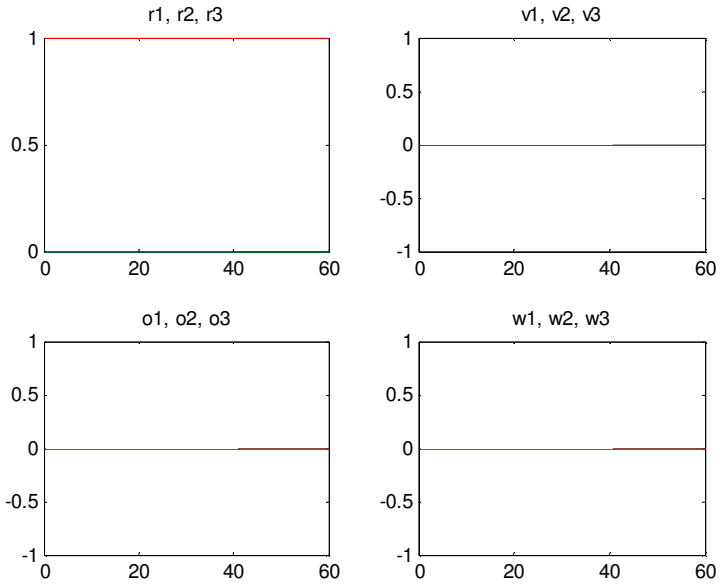
A program elején található inicializálás részben állíthatjuk be a kívánatos kezdetiérték feltételeket, valamint a szenzorok szimulálásához szükséges paramétereket (zaj, levágási frekvencia). Az autopilótának tetszőleges algoritmust kitalálhatunk, az én megvalósításomban ez egy egyszerű visszacsatolás, amely a kitérítő erővel ellentétes visszaszabályzást eszközöl.

6. Validálás, eredmények

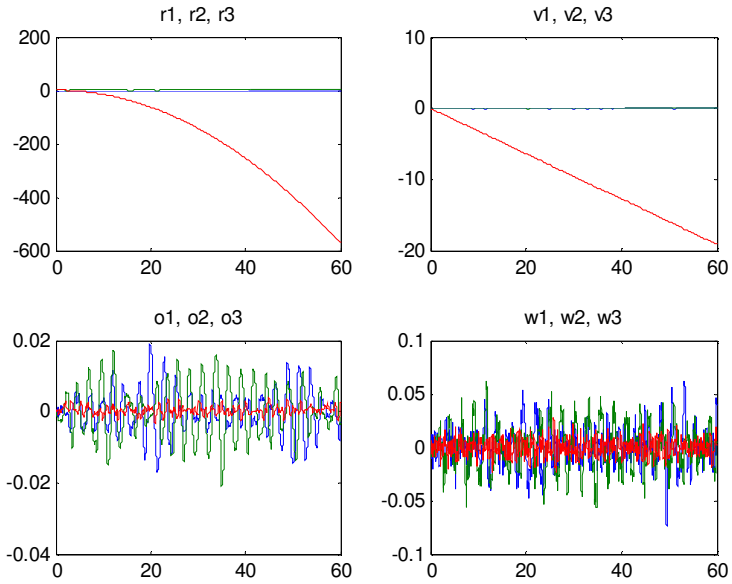
A szimulációs algoritmus megfelelőnek tekinthető, azzal ugyanis egyszerű teszteket hajtottam végre (egyetlen erő hatására történő gyorsítás, forgatás), melyek a törvények és a józan ész alapján várt eredményt szolgáltatták. Az autopilóta validálására sajnos még nem került sor, ahhoz még el kell készülni a quadrotor modell megépítésével, majd az autopilóta valós életben való teljesítésének az összehasonlításával. Az algoritmus egyszerűsége ugyanakkor bíztató jel arra nézve, hogy a valóságban is hasonlóan működne.

Incze Attila, AZ5FVR attila.incze@gmail.com

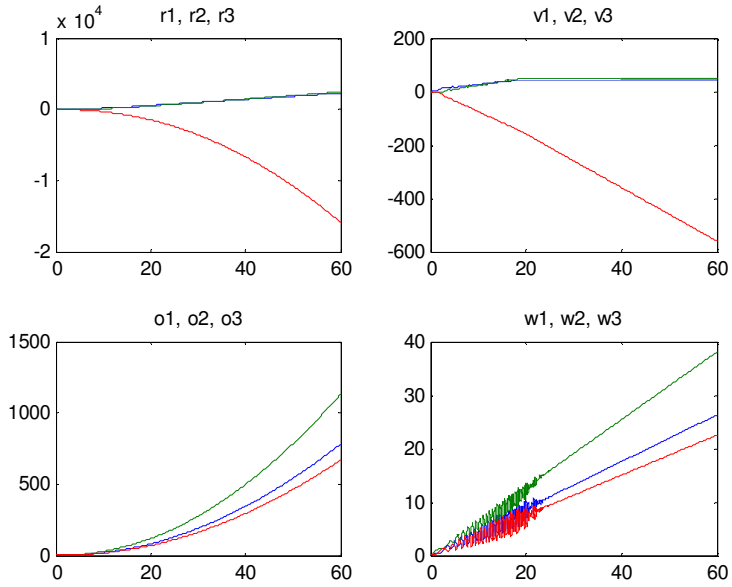
Az alábbiakban néhány tipikus eredményt mutatok be. A négy grafikonon az idő [s] függvényében láthatóak a helyvektor [m], sebesség [m/s], orientáció [rad], valamint szögsebesség [rad/s] értékek.



*A motorok **egyforma távolságra** vannak a tömegközéppontból, autopilóta **KI***
A test láthatóan teljes nyugalomban van. Természetesen ez az eset valóságban elérhetetlen.

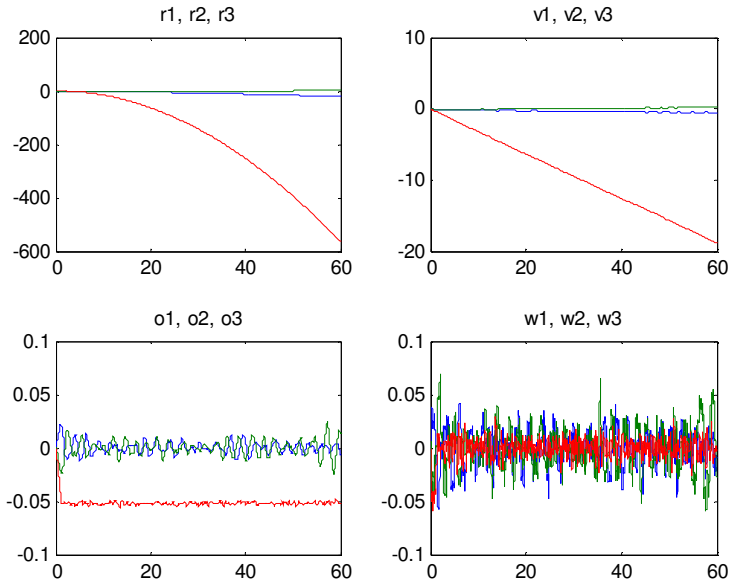


*A motorok **egyforma távolságra** vannak a tömegközéppontból, autopilóta **BE** (Autopilot v2.0)*
Nagyon kicsi oszcilláló mozgás: oka a szenzorok jelének a zaja. r3 csökken: lassan süllyed a test, mert az oszcillálás hatására a húzóerőnek van kicsi nem függőleges komponense.



*A motorok távolsága **nem** pontosan **egyenlő**, autopilóta **KI***

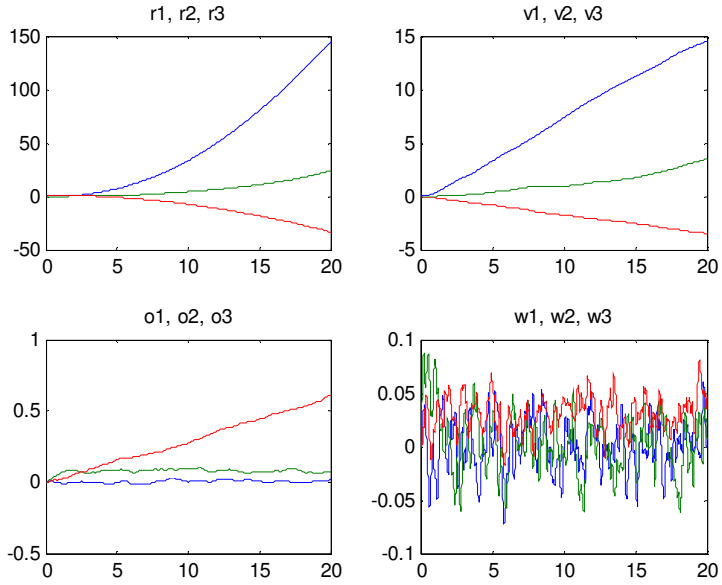
Jól láthatóan teljes káosz, sokszoros átfordulás minden tengely körül.



*A motorok távolsága **nem** pontosan **egyenlő**, autopilóta **BE** (Autopilot **v2.0**)*

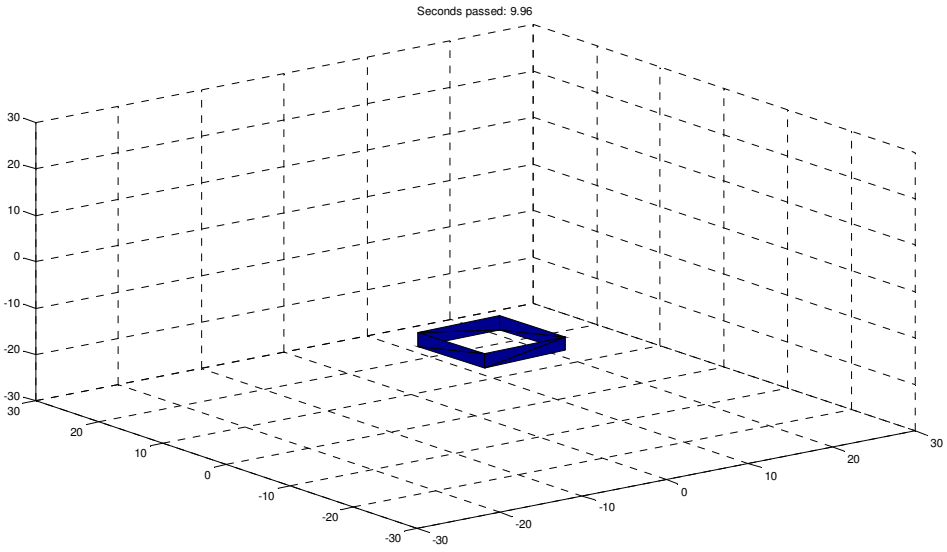
Stabilizált orientáció, és nagyon kis kimozdulás az x-y síkban. A test süllyedése természetesen itt is megfigyelhető. Kiküszöbölés megoldható a húzóerők kis növelésével. Összességében jó autopilóta algoritmus.

TTK, Fizikus MSc., Nappali
attila.incze@gmail.com



*A motorok távolsága **nem** pontosan **egyenlő**, autopilóta **BE** (Autopilot **v1.0**)*

Ez egy egyszerűbb autopilóta, mely csak az előző pillanatbeli szögsebesség alapján dönt, ezért csak a szögsebességeket tudja nulla közelében tartani, de az orientáció láthatóan folyamatosan változik. Az előző pontban látott v2.0 egy integrált szögesebesség értékkel is korrigál, mely információt hordoz a közeli múltról is, nem csak a jelenről.



A program futása végén egy ilyen 3D animációt láthatunk a test (helikopter) időbeli mozgásáról. Ilyen mód szemléletesen látható a test elmozdulása, és forgása.

7. Konklúzió

A szimuláció során tehát egy test mozgását szimuláltuk a 3D térben, megadott kezdőfeltételek, külső hatások, valamint visszacsatoló/szabályzó rendszer mellett.

A szimulátor abszolút eredményesnek tekinthető abból a szempontból, hogy sikerült olyan autopilóta algoritmust, valamint ahhoz megfelelő paramétereket találni, mely stabilizálja a helikopter mozgását. A kikísérletezett algoritmus és paraméterek ismeretében a modell helikopterre egyből egy jól működő stabilizáló rendszert implementálhatunk.

8. Hivatkozások

Rigid Body Dynamics, Chris Hecker - http://chrishecker.com/Rigid_Body_Dynamics