EÖTVÖS LORÁND TUDOMÁNYEGYETEM
TERMÉSZETTUDOMÁNYI KAR

Térképi vetületek 3D-s interaktív vizualizációja

SZAKDOLGOZAT
FÖLDTUDOMÁNY ALAPSZAK

Készítette:
Bokor Balázs
térképész és geoinformatikus szakirányú hallgató

Témavezető:
Dr. Györffy János
egyetemi docens
ELTE Térképtudományi és Geoinformatikai Tanszék

Budapest, 2011. május 11.

Tartalomjegyzék

1. A szakdolgozat témája és motivációja

A szakdolgozatom 3D-s interaktív programok létrehozásáról szól, melyeknek témája a földi és térképi koordináta-rendszerek. Ez a témakör a különböző rendszerek és az azok közötti kapcsolatok megalkotását, elemzését és rendszerezését foglalja magába. Az én célom az, hogy ehhez a témakörhöz hozzájáruljak olyan segédanyagokkal, melyek a témakörben szereplő mennyiségeket és definíciókat szemléltetik. Véleményem szerint ez a témakör az ezt befoglaló térképészet-geoinformatika-geodézia tudományterülettel együtt átlagosan elég alultáplált korszerű vizuális segédanyagokban összevetve más tudományterületekkel (pl.: gépészet, építészet, energetika, fizika, matematika, kémia...). Ezt a hiányt szeretném valamelyest csökkenteni úgy, hogy manapság már elvártnak tekinthető 3D-s és interaktív környezetben szemlélhető dinamikus ábrákat hozok létre.

Az ötlet a Földi és térképi koordináta-rendszerek című kurzus teljesítése közben született meg, ahol a megfelelő magyarázat mellett is sokszor csak viszonylag későn értettem meg a dolgokat, és sokat segített volna néha egy ilyen segédanyag. Természetesen egy ilyen segédanyag nem váltja ki a kapcsolódó témakör szóbeli és írásbeli magyarázatát, de meglehetősen jól kiegészíti.

2. Az elkészített programok és szerkesztésük bemutatása

A munka folyamán kilenc darab egymástól független program készült el. Olvasás közben érdemes elindítani őket, mert így kézzelfoghatóbb és élvezetesebb az olvasmány, és főleg azért, mert az elvégzett munka ezen programok használhatóságára irányult. Ez a fejezet egy általános és technikai ismertető után egyenként bemutatja a programok témáját, megértéséhez szükséges fogalmakat, majd a szerkesztés | programozás menetét, problémáit. A programozás bemutatásakor az elvégzett műveleteket nagyrészt hagyományos matematikai formátumban prezentálom annak érdekében, hogy mindenkinek érthető legyen, ne csak annak, aki ismeri a *Mathematica* nyelvezetét, de így is találkozni fogunk néhány paranccsal. A dolgozat elkészítéséhez szükséges munka oroszlánrészét a programozás és matematikai műveletek alkalmazása tette ki. Ez az írott tanulmányban is tükröződik valamennyire, így azt főleg az ezekben járatos olvasók találhatják majd érdekesnek. Viszont maguk a programok használatát remélem ennél több ember is élvezni fogja.

2.1. Általános ismertető

2.1.1. Felhasznált programok, technikai felépítés

Nem volt könnyű megtalálni a megfelelő módszert arra, hogy hogyan jelenítsem meg ötleteimet. Több szempontot állítottam fel, amelyek alapján elkezdtem keresgélni a programok között:

- ✔ Keltsen a megjelenítés háromdimenziós hatást, mivel ez a témakör görbült felületeken definiált koordináta-rendszerekről és azok síkba fejtéséről szól. Ráadásul manapság már minden témában láthatunk 3D-s vizualizációt.
- ✔ Legyen a megjelenítés elérése könnyű, vagyis viszonylag gyorsan betöltődjön, ne kelljen nagyméretű állományokat letölteni vagy külön programokat, beépülőket telepíteni, és ne kelljen szuper-számítógéppel rendelkezni futtatásukhoz.
- ✔ Kezelése könnyű, illetve könnyen tanulható, ezáltal felhasználóbarát legyen.
- ✔ A témához elegendő esztétikai követelményeket kielégítsen (pl.: színezés, megírás lehetősége).
- ✔ A témához elegendő pontossági követelményeket kielégítsen (pl.: lebegőpontos szám használata).
- ✔ A témához elegendő számítási követelményeket kielégítsen (pl.: nagy mennyiségű

számítás, beépített függvények, dinamikusság).

- ✔ Interaktív legyen, tehát a felhasználó tudjon kommunikálni a programmal, és ne csak statikus ábrákat lássunk.
- ✔ És nem utolsó sorban az, hogy általam kivitelezhető legyen.

A feladat megoldásához több programot kipróbáltam: GeoGebra3D/2D, ArchimedesGeo3D, Geomview, K3DSurf, The Geometer's Sketchpad, Blender... Továbbá megvizsgáltam nagyobb számítógépes algebra rendszerek alkalmazásait (CAS), mint a Maple, MathCad, MATLAB, Origin, de végül egy vektoranalízis oldalt böngészve találtam meg amit kerestem [Nykamp, 2009]. Ez egy weboldalba ágyazott Java applet, ami a 2.1.3.-ben leírtak szerint viselkedik. Szerkezetét tekintve úgy néz ki, hogy a program forráskódja a weboldalba van beágyazva (vagy hivatkozunk rá), viszont ez nem *Java* nyelven íródik, hanem egyéb nyelven. Ezt a még be nem mutatott nyelvet a *Java* futtatókörnyezet nem tudja olvasni, ezért egy tömörített fordítófájl átalakítja a *Java* számára, amely már így le tudja futtatni a programot. Tehát röviden: Mi megírjuk a programunkat a .html fájlunkba egy bizonyos nyelven, amit egy hivatkozott fordítócsomag emészthetővé alakít a *Java* motor számára, s az már tudja is futtatni az applet-et. Ez a csomag már biztosít nekünk előre egy környezetet (grafikus motor, 3D-s hatás, forgatások -, interakció -, szövegkezelés, stb..), amiben mi (betartva a szabályokat és a korlátokat) már alkothatunk is. A csomag neve *LiveGraphics3D*(**live.jar**), készítője Prof. Dr. Martin Kraus német egyetemi kutató|oktató, programozó informatikus|matematikus. Körülbelül 100 KiB-nyi helyet foglal el, amit a böngésző automatikusan letölt. A .html fájl a következőképpen épül fel: Az „**<applet>**”és „**</applet>**” jelzők közt található a program forráskódja. Ezen belül először hivatkozunk a fordítócsomagra az **archive=”...”** és **code=”..”** utasításokkal, majd megadjuk az applet méreteit, amiket én 1024*768-as képernyőfelbontásra optimalizálva 1000*600-asnak választottam. Ezek után különböző paraméterek értékeit adjuk meg ilyen formában: **<param name=... value="...">**. Ilyen paraméter lehet például a háttérszín, a háttérkép, a nagyítás kezdőértéke, a nézőpont keringésének szögsebessége, hanglejátszás, ...stb. Ezek közül a három legfontosabb paraméter az **independent_variables**, a **dependent_variables** és az **input**. Az érdemi munka nagy része ezeknek a definiálására irányult. Az első két paraméter értékének felsorolásszerűen értékadó utasításokat adunk meg, viszont a kettő között annyi a különbség, hogy míg a független változóknak majd az egér által interaktívan, vagy az idő telésével adunk értéket, addig a függő változókban csak a tőlünk

nem, vagy csak közvetve függő értékek szerepelhetnek. Tehát az elsőben szerepelnek azok az értékek, amelyek a felhasználó közvetlenül módosíthat, a másodikban pedig amelyek ezektől nem feltétlenül függenek. A harmadik paraméter tartalmazza azokat a parancsokat, amelyek a kirajzolásért felelnek. Például pontok, vonalak, sokszögek, szövegek koordinátáit, és formázási tulajdonságait adhatjuk meg, ahol felhasználhatjuk a fentebb kiszámított értékeket. Szerencsére van lehetőségünk az **input** paraméterben szereplő parancsok összességét betömöríteni egy .zip fájlba, és mivel terjedelmét tekintve ez a paraméter teszi ki az elkészített programok felemésztő részét, igencsak lecsökkenti a majd letöltendő fájlok méretét (így az applet-eim összmérete 17-ről 2.7 MiB-ra csökkent). Erre a .zip fájlra és az azt tartalmazó szövegfájlra a **<param name=input_archive value="....zip">** és a **<param name=input_file value="....***kiterjesztés***">** parancsokkal hivatkozhatunk. A dolgozatban használt szövegfájlok kiterjesztései Bajnai Sándor orgonaművészről kapták neveiket (**.sanyi**). Az applet-nek keretet is csináltam, amelyet úgy értem el, hogy azt egy táblázat (**<table>**) egy sorának (**<tr>**) mezejébe (**<td>**) helyeztem el, és meghatároztam a tulajdonságait.

2.1.2. Forráskód szerkesztése

A szakdolgozat elkészítéséhez *Notepad++*-t mint szövegszerkesztőt; *Opera*-t mint böngészőt, *Total Commander*-t mint fájlkezelőt és tömörítőt, *Java*-t mint futtatókörnyezetet, *LibreOffice*-t mint dokumentum-, táblázat- és prezentációszerkesztőt, és *Mathematica*-t mint műveletvégző és szöveggeneráló számítógépes algebra rendszert használtam. A dolgozatban leírt műveleteket *Mathematica*-ban végeztem el, amely egy hihetetlenül erős alkalmazásnak bizonyult használata során. Minden műveletet igény szerint el tudtam végezni az adatsor beolvasástól és kezeléstől kezdve vektor- és, differenciálszámításon, egyenletek megoldásán át többváltozós függvények vizualizációjáig és sorozatgenerálásig, és még sorolhatnám, hogy meddig. És én csak a rendszer egy apró szegmensével találkoztam. Minden műveletnél az összes megoldást megadta, és ha valami nem jött ki, biztos lehettem benne, hogy én hibáztam. Felhasználói felülete parancs alapú. Hihetetlenül dinamikus tud lenni, feltéve, ha közölni tudjuk vele amit akarunk. Azért szükségeltetett a *Mathematica* használata, mert a görbéket alkotó nagyon sok pontot kézzel nem tudom számítani és bevinni, kellett egy szöveggeneráló alkalmazás. Másrészt mert ennek a programnak a nyelvét (vagyis csak egy kis részét) ismeri fel a *LiveGraphics3D*. Mivel a *LiveGraphics3D* fordítócsomag nem görbéket rajzol, hanem vonal-szakaszokat, ezért a megjelenítendő görbéket sok apró szakaszból állítottam össze. Ezt

úgy értem el, hogy vettem a görbe paraméteres egyenletrendszerét és behelyettesítettem a paraméter helyébe egy értéket, így kaptam egy koordináta-hármast. Ezután a paraméter helyébe egy kicsit más értéket tettem, majd ezt az előző mellé tettem. Így keletkezik egy koordinátasorozat, amelyet betettem a vonalat kirajzoló parancs (**Line[..]**) argumentumába. Ezt a ciklikus műveletet egy **for**-szerű paranccsal, a **Table[..]**-lel automatizáltam. Tehát ebbe a parancsba a kívánt görbét betéve megkapom az azt közelítő polyline-t, amelyet azután az **input** paraméterbe helyezek. Ilyen módon csináltam meg az összes görbét és sokszöget az elkészített programokban.

Szerkesztés során vizsgálgattam más által elkészített programokat, amelyeket a Dr. Kraus honlapján, és egy másik ehhez kapcsolódó honlapon találtam (Rogness).

2.1.3. Használati útmutatás

A programokat a mellékelt .html fájlok böngészőben való elindításával érhetjük el amennyiben a számítógépünkön telepítve van a *Java* futtatókörnyezet, és az a böngészőben megfelelően lett konfigurálva. Ekkor megjelenik előttünk egy kerettel szegélyezett ablak, amelyben (ha minden jól működik,) megjelenik a fordítócsomag logója, amíg betöltődik az alkalmazás. Ha betöltődött, a kereten belül megpillantunk egy háromdimenziós helyszínt, amelyben helyet kapnak az általam megrajzolt objektumok. Ebben a térben elhelyezkedő nézőpontunkat intuitíven forgathatjuk egy megszabott pont körül, nagyíthatunk, kicsinyíthetünk, a nézőpont távolságát állíthatjuk, időben változó objektumok időváltozóját változtathatjuk (tehát animációt tekerhetünk), de ami a legfontosabb, hogy az objektumok paramétereit állíthatjuk kitüntetett pontok segítségével. Itt a teljesség igénye nélkül felsorolom az irányítás módjait, hogy a dolgozat olvasása közben tudjon szemlélődni az olvasó:

Bal kattintás és mozgatás	A nézőpont helyzetének forgatása
Bal kattintás egy ponton és mozgatás	A pont helyzetének változtatása
Shift + bal kattintás + függőleges mozgatás	Nagyítás állítása
Shift + bal kattintás + vízszintes mozgatás	Nézőpont vektora körüli forgatás
Ctrl + bal kattintás + függőleges mozgatás	Nézőpont távolságának állítása
Dupla bal kattintás	Animáció megállítása és elindítása
Jobb kattintás + vízszintes mozgatás	Animáció tekerése
Home	Kezdő értékek visszaállítása

2.2. Az applet-ek

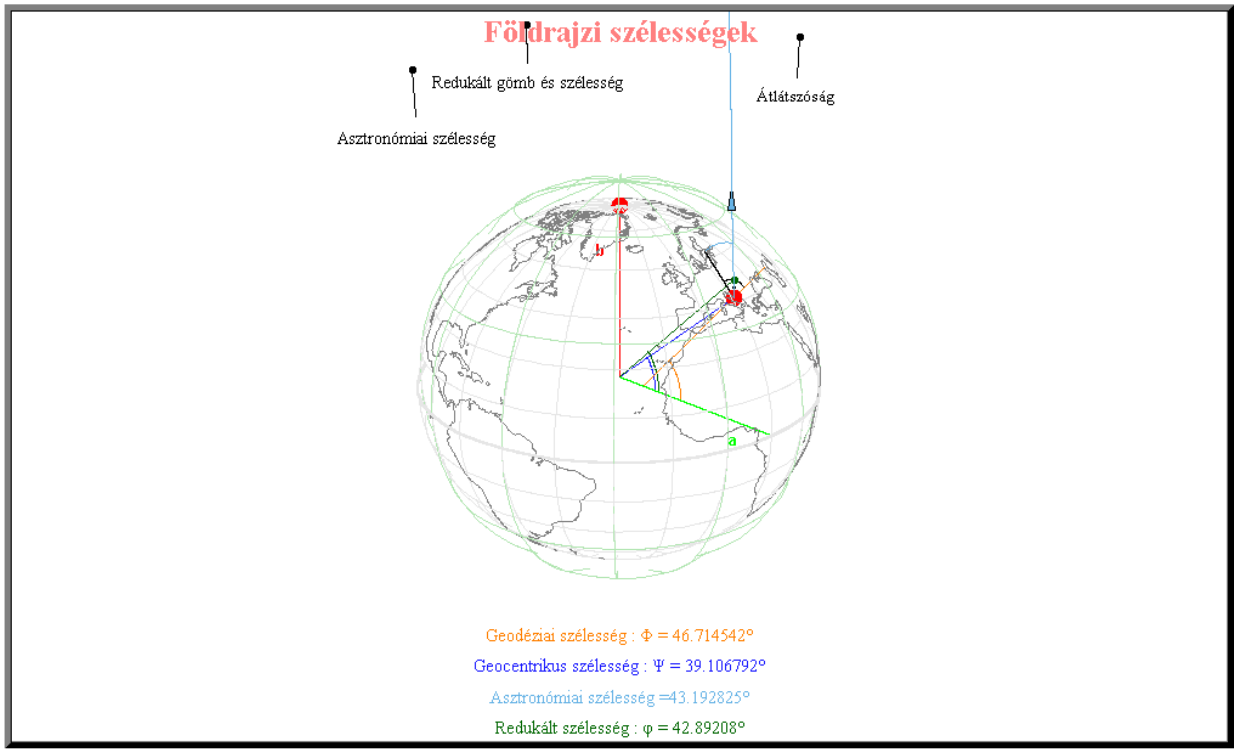
A programok témájukat tekintve megpróbálják a térképkészítés procedúrájának egy fontos szakaszát felölelni. Ez a szakasz ott kezdődik, hogy a geodétáktól, vagy más felmérésből származó ellipszoidi koordinátákat egy vonatkoztatási rendszerben megkapjuk. Ezeket utána átalakítjuk gömbfelületi koordinátákra, amelyeket már gömbháromszögtani módszerekkel átszámolhatunk másik tetszőleges gömbfelületi koordináta-rendszerbe. Ezt az átalakított koordináta-rendszert használhatjuk ezután a gömbfelület síkba fejtéséhez. Mint tudjuk vetülettanból, egyáltalán nem csak ilyen kettős vetületek léteznek, de az applet-ek csak ezekre korlátozódnak. Tehát az adott forgási ellipszoid alapfelületi pontoknak megkapjuk térképi helyeit, így ezek készek a további, már síkbeli feldolgozásra. Az elkészített programok által megjelenített tudásanyag nagyon nagy része a témavezetőm által tartott órákról és az internetes jegyzetéből származik, a többi pedig (pl.:2.2.5., 2.2.2.-ben a simulógömb,...) tőlem, vagy egyéb internetes forrásokból (2.2.6.: Wikipédia).

2.2.1. Földrajzi szélességek

Az első program szeretné láttatni Földünk többedleges közelítésének számító valamelyik forgási ellipszoidnak, mint felületnek az egyik természetes paraméterét, a szélességet. Szemben a másik paraméterrel, a hosszúsággal, a szélességet többféleképpen lehet definiálni. Az egyik földrajzi szélesség a geocentrikus szélesség, amely egy adott ellipszoidi felszíni pont és a felület geometriai középpontja által meghatározott egyenes és az egyenlítői sík által bezárt szög. Ezt műholdas alkalmazásoknál használják. Másik meghatározás már az ellipszoid felületének normálisát alkalmazza, miszerint az azt tartalmazó egyenes és az egyenlítői sík által bezárt szög adja a geodéziai szélességet. A harmadik szélesség a redukált szélesség, amely hasonló a geocentrikus szélességhez, csak nem a felületi ponthoz húzunk egyenest a középpontból, hanem a felületi pont **Z** tengely irányába, egy **a** sugarú gömbre való vetítésekor keletkezett ponthoz. Létezik még több fajta szélesség (pl.:izometrikus szélesség, amelyet gömb vagy forgási ellipszoid szögtartó leképezéseihez (pl.:GK,UTM) is használunk), de a program csak ezt a hármat tartalmazza. Tehát egy pontnak az ellipszoidon többféle koordinátapárja is lehet. E három között az átszámítás viszonylag egyszerű, a szerkesztésnél meg is lesz adva. Az applet tartalmaz még plusz egy szélességet, amelynek megjelenítése csak félig-meddig takarja a valóságot, ezért elővigyázattal használjuk. Ez a csillagászati szélesség.

Ez úgy jelenik meg a programban, hogy az ellipszoid pontbeli meridián irányú érintője és a Sarkcsillag (Polaris) iránya szögszárakat alkotnak, és közöttük egy körív húzódik. Azért inkorrekt a megjelenítése, mert annak a definíciója a földi nehézségi erőtér egyik szintfelületén a gradiensvektor negatívjának és egy meghatározott irányra (pl.:ITRS Reference Pole) merőleges síkkal bezárt szöge, tehát itt már nem a program által megjelenített forgási ellipszoidról van szó, hanem valamely ekvipotenciális szintfelületről [Biró, 2004]. A korrekt megjelenítésbe nem fektettem kellő energiát, de biztos, hogy egy alacsony fokú és rendű gömbfüggvény sorbafejtésével ki lehet alakítani egy nem feltétlenül élethű, de szemléletes felületet (krumpli), amelyen pontunkat mozgatva helyesen látszódna ez a fajta szélesség.

A programban egy változtatható lapultságú forgási ellipszoidot láthatunk, amelynek felületén egy pontot mozgathatunk. Erre a pontra vonatkozóan megjelennek a koordináták értelmezései és értékei. A letisztult kép érdekében kapcsolókkal irányíthatjuk, hogy éppen mi jelenjen meg.



1. ábra

- **Szerkesztés**

Legelőször megcsináltam az applet vázát a 2.1.1.-es bekezdés szerint. Utána elkészítettem a

forgási ellipszoidot. Felületről lévén szó megtehettem volna, hogy szegélyezett vagy szegélyezetlen sokszögekből építem fel, de mivel a test belsejében is van megjeleníteni való objektum, ezért a láthatóság kedvéért a test 15°-os drótvázát készítettem el. A meridiánokat és a parallelköröket egy-egy kétszintű **Table[..]** függvénnyel generáltam 2.1.2. szerint, a következő **u** és **v** paraméterű egyenletrendszert használva:

$$\vec{\mathbf{E}}(\mathbf{u},\mathbf{v}) := \begin{bmatrix} \mathbf{a}\cdot\cos\left(\arctan\left(\frac{\mathbf{b}}{\mathbf{a}}\cdot\tan(\mathbf{u})\right)\right)\cdot\cos(\mathbf{v}) \\ \mathbf{a}\cdot\cos\left(\arctan\left(\frac{\mathbf{b}}{\mathbf{a}}\cdot\tan(\mathbf{u})\right)\right)\cdot\sin(\mathbf{v}) \\ \mathbf{b}\cdot\sin\left(\arctan\left(\frac{\mathbf{b}}{\mathbf{a}}\cdot\tan(\mathbf{u})\right)\right) \end{bmatrix} \quad \textbf{ahol} \quad \begin{matrix} -90° \le \mathbf{u} \le 90° \\ -180° < \mathbf{v} \le 180° \end{matrix}$$

Látható, hogy az elkészített paramétervonalak a geodéziai szélességet használják, tehát ha az ellipszoid lapultságát változtatjuk, akkor a szélességi vonalak eltolódnak a felületen. Viszont ehhez még meg kell adni **a**-t és **b**-t. **a**-nak egyszerűen értéket adtam a **dependent_variables** paraméterben, **b**-t viszont a felhasználó állíthatja be. Ezért az **independent_variables** paraméter értékébe beírtam **b**-t, a lapultságot állító fogantyú egy paraméterét, amely egyben az ellipszoid kis féltengelye is, majd ezt az **input**-ban egy pont koordinátájaként felhasználtam (**Point[{0,0,b}]**).

Itt kell megemlítenem az interakció működési elvét, amelyet az összes applet használ. Ha az **input**-ban szereplő valamelyik **Point[..]** koordinátái közül legalább az egyik független változó, akkor azt a pontot kattintással és húzással áthelyezhetjük. Kattintáskor a pontot csak a nézőpont-vektorra merőleges síkon tudjuk mozgatni az egérrel, viszont meg lehet oldani, hogy az mégis egy tetszőleges térbeli alakzaton belül helyezkedjen el (pont, görbe, felület, test). Ezt úgy teszem, hogy az egér által megadott érték megjelenik az **independent_variables** szekcióban, de azt újbóli értékadással a **dependent_variables** szekcióban módosítom, így a kirajzolandó **input**-ba bemenő adat már az új érték lesz. És ez minden ciklusban megismétlődik. Így tudom megoldani, hogy ezen a síkon mozgó pontot minden ciklusban „odateszem” a nekem megfelelő helyre, például az ellipszoid felszínére, vagy egy kapcsoló két végpontjába.

Az ellipszoid lapultságát 0 és ½ között változtathatjuk, így szemléletesen változnak az ezeken alapuló értékek, mint például a szélességértékek különbsége, vagy a szélességi körök helyzete

az ellipszoidon.

Beírtam még az **independent_variables** paraméter értékébe a majd mozgatható felületi pont három koordinátáját is $\vec{P}=[x,y,z]$. Ugyanitt szerepelnek ezek kezdőértékei is, amelyek a legelső ciklusnál használatosak betöltődéskor. A koordinátákból kiszámoltam a három fajta szélességet:

Geocentrikus szélesség: $\mathbf{gcf}:=\arcsin\left(\frac{z}{\sqrt{x^2+y^2+z^2}}\right)=\Psi;$

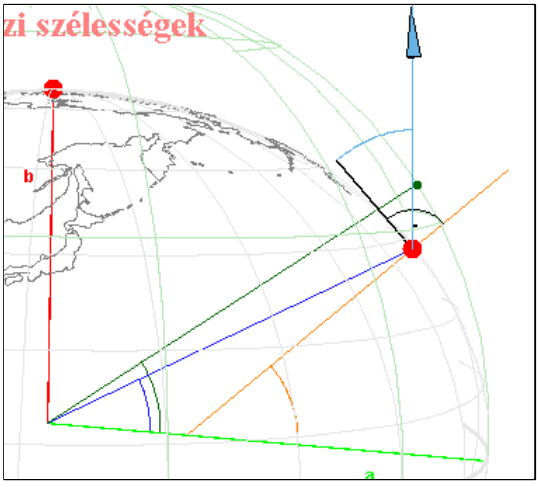
Redukált szélesség: $\mathbf{rf}:=\arctan\left(\tan(\Psi)\cdot\frac{a}{b}\right)=\varphi;$

Geodéziai szélesség: $\mathbf{gdf}:=\arctan\left(\tan(\Psi)\cdot\left(\frac{a}{b}\right)^2\right)=\Phi$

A $\vec{P}$ pontból vonalat húztam az origóba, majd megszerkesztettem egy körívet, amely a geocentrikus szélesség szögét hivatott jelezni:

$$\vec{GC}(g):=\begin{bmatrix}a/4\cdot\cos(\Lambda)\cdot\cos(\Psi\cdot g)\\ a/4\cdot\sin(\Lambda)\cdot\cos(\Psi\cdot g)\\ a/4\cdot\sin(\Psi\cdot g)\end{bmatrix}\quad\text{ahol}\quad 0\le g\le 1$$

Ezután megcsináltam az **a** sugarú gömb zöld drótvázát, amelyen a redukált szélesség értelmezve van. Ezen egy $\vec{R}$ pont jelzi $\vec{P}$ pont egyenlítőre merőleges vetületét erre a gömbfelületre. Két másik körívet és vonalat szerkesztettem $\vec{GC}$-hez hasonlóan, felhasználva a másik két szélességet. Ezután berajzoltam a $\vec{P}$ -hez északi irányba egy derékszöget, jelezve a normális merőlegességét a felületre. Ezt egy ortodróma segítségével alkottam meg, amelynek szerkesztését 2.2.3.-ban részletezem majd. Az itt használt két vektor, amelyekkel az ortodrómát csináltam:



2. ábra

Északi irányú érintő: $\vec{\mathbf{tan0}}:=\begin{bmatrix}-\sin(\Phi)\cdot\cos(\Lambda)\\ -\sin(\Phi)\cdot\sin(\Lambda)\\ \cos(\Phi)\end{bmatrix}+\vec{P}$

Felületi normális: $\vec{\mathbf{norm}} := \begin{bmatrix} \cos(\Phi)\cdot\cos(\Lambda) \\ \cos(\Phi)\cdot\sin(\Lambda) \\ \sin(\Phi) \end{bmatrix} + \vec{\mathbf{P}}$

Az applet leírásánál megemlített plusz egy szélességnek a szerkesztését a Sarkcsillag (**=[0,0,pbz]**) megjelenítésével kezdtem, majd kiszámoltam a felületi pontnak a csillagászati szélességét:

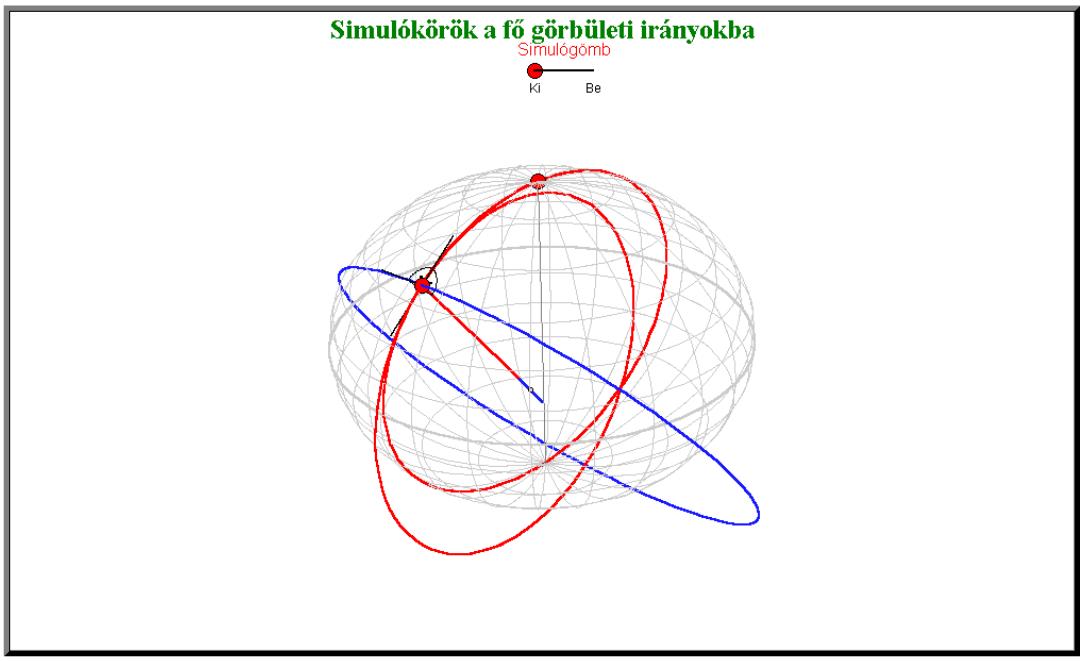
$$\mathbf{asztf} := \arctan\left(\frac{\mathbf{pbz}-\mathbf{z}}{\sqrt{\mathbf{x}^2+\mathbf{y}^2}}\right)+\Phi-\mathbf{90}^\circ$$

Mind a redukált-, mind a csillagászati szélességet egy-egy kapcsolóval kikapcsolhatjuk, hogy átláthatóbb legyen az ábra. Ezt úgy működik, hogy egy kapcsolónak a végpontjaihoz rendelek két értéket (0 és 1), és ezt az értéket vizsgálja egy **HA()** függvény, hogy az adott objektumot megjelenítse, vagy sem. A kiírt értékek, jelölések és görbék között színkód teremt kapcsolatot, tehát mindegyik fajta szélességhez egy-egy szín tartozik.

Sok applet-ben megjelennek kontinensek kontúrvonalai. Ezek a témavezetőtől kapott **FOLD_jav.txt** nevű koordinátasorozatból származnak, amely mintegy ~14 000 koordinátapárt tartalmaz tizedfok pontossággal. Ennek a felbontása teljesen megfelel az applet-ekben való szemlélődéshez, viszont a belőlük generált koordinátahármasok méretben bőven felülmúlják az applet-ekben szereplő többi görbének a koordinátáit. Hogyha meg mindegyik koordinátában még elvégzendő művelet is található, akkor a processzorhasználat is eléggé megnő. Például a 2.2.7.-ben lévő alapfelületi kontinensvonalak ~900 KiB méretűek. Itt minden „sziget” polyline formájában jelenik meg, tehát minden adat csak egyszer szerepel. Viszont a többi programban a kontinensek polyline-jait szétdobtam apró line-okra, nagyjából megduplázva a kód méretét és a számítások számát. Mielőtt ezt a sületlenséget látva szívrohamot kapna a topológiát tanult olvasó, megjegyzem, hogy ez szükséges. Hogy a Föld velünk átellenben lévő oldalán fekvő vonalakat eltakarjam, egy nagy fehér pontot tettem az origóba (ennek a pontnak a jelenlétét állíthatjuk az „Átlátszóság” kapcsolóval). Viszont a program takarás esetén a közelebbi objektumot jeleníti meg, így egy polyline esetén, amelynek egyik végének esetleg takarásban kéne lennie, teljesen megjelenik vagy eltűnik, összezavarva a képet. Tehát a takarás miatt kell majdnem minden objektumot ilyen módszerrel szétdarabolni, a forráskód és a számítások rovására. Ezekből látható, hogy a futás közbeni számítások nagy részéért a kontinens kontúrvonalai a felelősök.

2.2.2. Forgási ellipszoid görbületi viszonyai

A második program bemutatja a forgási ellipszoid alapfelületű vetületekhez és a felületen való egyéb számításokhoz szükséges alapmennyiségeket. Ezek név szerint: meridiángörbületi- és harántgörbületi sugár. Ha egy analitikus felületen kijelölünk egy pontot, akkor ott a felületi normálist tartalmazó síkok és a felület metszésvonalaihoz simulóköröket illeszthetünk változó nagyságú sugárral. E sokaságnak a szélső értékeit fő görbületi sugaraknak hívjuk. Ezt alkalmazhatjuk egy forgási ellipszoidra is, ahol azt látjuk, hogy egy tetszőleges pontban a legkisebb görbületi sugarat (vagyis a legnagyobb görbületet) meridián-irányban, a legnagyobb görbületi sugarat (vagyis a legkisebb görbületet) az arra merőleges irányban találjuk. (Ezeknek a simulóköröknek a segítségével tudunk például hosszakat mérni az ellipszoid felszínén. Mondjuk egy meridiánív hosszát úgy tudjuk megmérni, hogy a szélesség függvényében pontról pontra változó sugarú meridián irányú simulókörnek egy végtelenül kicsi ívhosszát pontról pontra kiszámítjuk, majd ezeket összegezzük. Másik alkalmazásuk például, hogy a különböző forgási ellipszoid alapfelületű vetületek levezetése kezdetén igény szerint kapcsolatot teremtünk az alapfelület és a képfelület között, amely kapcsolat matematikai leírása egy görbületi sugarakat tartalmazó differenciálegyenlet lesz.) Ezen simulókörök térbeli helyzete, sugaruk nagysága, és azon görbék szemléltetése a cél, amelyekhez hozzásimulnak.



3. ábra

- **Szerkesztés**

Az előző applet-ből átmásoltam az ellipszoidot megjelenítő vonalakat, a lapultságot állító pontot és a mozgatható pontot a felületen, viszont a kontinensek körvonalait nem. Utána létrehoztam a harántmetszet simulókörét és sugarát, ami kék színezetet kapott. Ehhez tudnom kellett a görbének a paraméteres egyenletrendszerét. Először meghatároztam a kör középpontját, ami egy egyszerű derékszögű háromszögből jött ki. Átfogója: $\mathbf{N}(\mathbf{\Phi})=\frac{\mathbf{z}}{\sin(\Phi)}\cdot(\frac{\mathbf{a}}{\mathbf{b}})^2$; egyik befogója ami egyben a parallelkör sugara: **r(Φ);** három csúcsa: 1. a mozgatható felületi pont $\vec{\mathbf{P}}=[\mathbf{x},\mathbf{y},\mathbf{z}]$; 2. annak a forgástengelyre (**Z**) való vetülete **[0,0,z]**; 3. a kör középpontja a tengelyen ismeretlen **Z** koordinátákkal $\vec{\mathbf{Ncent}}=[\mathbf{0,0},\mathbf{Ncentz}]$. Onnan tudom, hogy az középpont a tengelyen helyezkedik el, mert Meusnier tétele kimondja:

$$\mathbf{r}(\Phi)=\mathbf{N}(\Phi)\cdot\mathbf{cos}(\Phi)$$

Amiből látható, hogy a parallelkör sugara a $\vec{\mathbf{P}}$ pontból induló harántgörbületi sugár **XY** síkbeli komponense, és mivel a parallelkörök középpontja a forgástengelyen van, ezért a harántmetszet simulókörének is. Ebből a háromszögből **Φ**-t az előző applet-ben leírtak alapján számítom, a maradék befogót pedig $\mathbf{N}(\Phi)\cdot\mathbf{sin}(\Phi)$ alapján. Ebből kivonva $\vec{\mathbf{P}}$ pont **Z** koordinátáját megkapom a középpont **Z** koordinátáját. Tehát $\mathbf{Ncentz}(\Phi)=\mathbf{z}-\mathbf{N}(\Phi)\cdot\mathbf{sin}(\Phi)$. A görbe meghatározásának második lépése, hogy a $\vec{\mathbf{P}}$ pontot képzeletben az $+\mathbf{X}\pm\mathbf{Z}$ félsíkon helyezem el. Ekkor a **g** paraméterű görbe egyenletrendszere:

$$\vec{\mathbf{K}}(\mathbf{g}):=\begin{bmatrix}\mathbf{r}(\Phi)\mathbf{cos}(\mathbf{g})\\ \mathbf{N}(\Phi)\mathbf{sin}(\mathbf{g})\\ \mathbf{N}(\Phi)\mathbf{sin}(\Phi)\mathbf{cos}(\mathbf{g})\end{bmatrix}$$

Ez viszont csak a pontunk szélességétől függ, hosszúságától nem, ezért végezetül megszoroztam balról egy **Z** tengely körüli **Λ** szerinti jobbsodrású forgatómátrixszal, majd eltoltam az origóból:

$$\vec{\mathbf{H}}(\mathbf{g}):=\vec{\mathbf{R}}_{\mathbf{Z}}(\Lambda)\cdot\vec{\mathbf{K}}(\mathbf{g})+\vec{\mathbf{Ncent}}=\begin{bmatrix}\mathbf{r}(\Phi)\mathbf{cos}(\mathbf{g})\mathbf{cos}(\Lambda)-\mathbf{N}(\Phi)\mathbf{sin}(\mathbf{g})\mathbf{sin}(\Lambda)\\ \mathbf{r}(\Phi)\mathbf{cos}(\mathbf{g})\mathbf{sin}(\Lambda)+\mathbf{N}(\Phi)\mathbf{sin}(\mathbf{g})\mathbf{cos}(\Lambda)\\ \mathbf{N}(\Phi)\mathbf{cos}(\mathbf{g})\mathbf{sin}(\Phi)+\mathbf{Ncentz}(\Phi)\end{bmatrix}$$

Ezt használtam fel egy **Table[...]** függvényben, amely egy listát generált, változónak tekintve a **g** paramétert. Azután egy másik görbét, a meridián simulókörét hoztam létre, amit piros

színnel jelöltem. Ehhez kell az ellipszoid excentricitásának a négyzete, és a meridiángörbületi sugár:

$$e2 := \frac{a^2 - b^2}{a^2} \qquad M(\Phi) := \frac{a(1-e2)}{(1-e2\cdot\sin^2(\Phi))^{\frac{3}{2}}}$$

Itt is meghatároztam a kör középpontját:

$$\vec{Mcent}(\vec{P}) := \begin{bmatrix} x\left(1-\frac{M(\Phi)}{N(\Phi)}\right) \\ y\left(1-\frac{M(\Phi)}{N(\Phi)}\right) \\ z - M(\Phi)\cdot\sin(\Phi) \end{bmatrix}$$

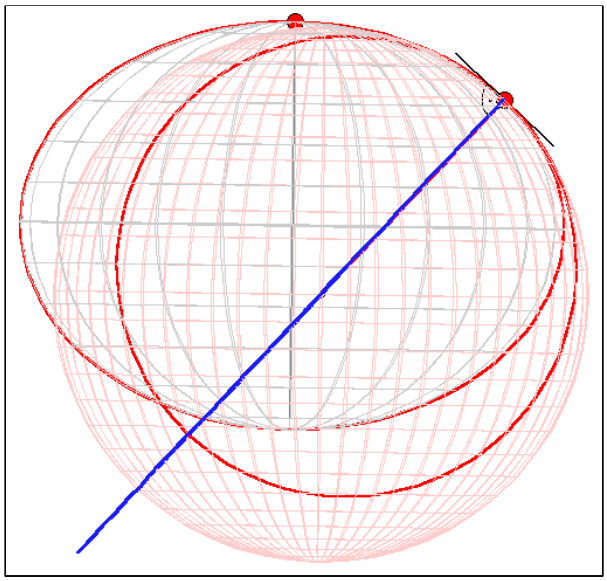
Majd ennek segítségével magát a kört:

$$\vec{L}(g) := \begin{bmatrix} M(\Phi)\cdot\cos(\Lambda)\cdot\cos(g) \\ M(\Phi)\cdot\sin(\Lambda)\cdot\cos(g) \\ M(\Phi)\cdot\sin(g) \end{bmatrix} + \vec{Mcent}$$

Létrehoztam magát a bimeridiánt is, amelyhez az utóbbi kör simul. Ez egyszerű volt:

$$\vec{B}(g) := \begin{bmatrix} a\cdot\cos(\Lambda)\cdot\cos(g) \\ a\cdot\sin(\Lambda)\cdot\cos(g) \\ b\cdot\sin(g) \end{bmatrix}$$

Végül egy 2.2.1.-ben lévőkhöz hasonló kapcsolót csináltam, hogy egy plusz objektumot megjeleníthessen a felhasználó kedve szerint. Ez az objektum a simulógömb, amely az adott $\vec{P}$ pontban a legjobban simul a forgási ellipszoidhoz. Ez a gömb a képfelülete a Gauss-féle szögtartó gömbvetületnek, amely az első fázisa egy olyan kettős vetítésnek, mint például az Egységes Országos Vetület. De van-e értelme egy olyan vetületnek az alap-, és képfelületét térben szemléltetni, ami nem is geometriai úton jött létre? Mivel ennek a gömbvetületnek a létrehozásakor többek között olyan szempontok szerepelnek, mint a szögtartóság és hossztartó parallelkörök közti kapcsolat, láthatjuk, hogy semmi sem garantálja a térbeli kapcsolatot. De léteznek azért ilyen



4. ábra

irányú kutatások [Kratochvilla, 2003]. Annak ellenére ha mégis létezne ilyen kapcsolat, elkészítettem a magam interpretációját, hogy legyen lehetőségünk szemlélni ezt a térbeli viszonyt. Viszonylag egyszerű dolgom volt, mivel a gömb középpontját $\mathbf{M}\vec{\mathbf{cent}}$-hez hasonlóan találom meg, kicserélve **M(Φ)**-t **R(Φ)**-re, ahol $\mathbf{R}(\Phi)=\sqrt{\mathbf{M}(\Phi)\cdot\mathbf{N}(\Phi)}$**.** Így simulógömbünk egyenletrendszere:

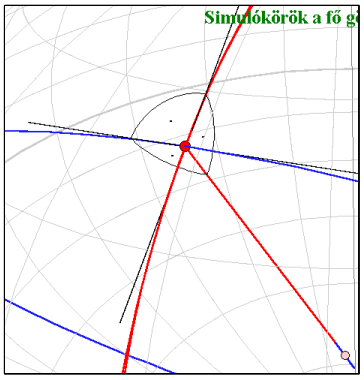
$$\vec{\mathbf{S}}(\mathbf{u},\mathbf{v}):=\begin{bmatrix}\mathbf{R}(\Phi)\cdot\cos(\mathbf{u})\cdot\cos(\mathbf{v})\\ \mathbf{R}(\Phi)\cdot\cos(\mathbf{u})\cdot\sin(\mathbf{v})\\ \mathbf{R}(\Phi)\cdot\sin(\mathbf{u})\end{bmatrix}+\mathbf{R}\vec{\mathbf{cent}}\quad\textbf{ahol}\quad\begin{matrix}-90°\le\mathbf{u}\le 90°\\ -180°<\mathbf{v}\le 180°\end{matrix}$$

Átláthatóság kedvéért a gömbnek áttetsző felületet szerettem volna adni, de mivel ez nem lehetséges ezzel a megjelenítési módszerrel, ezért máshoz folyamodtam. Sűrű és halvány drótvázát jelenítettem meg a felületnek. A drótváz vonalai a szokványos és legegyszerűbb paraméterek szerint húzódnak, nincs közük a fentebb említett gömbvetületi koordinátákhoz. Ha csak statikusan, képként vizsgáljuk a programot, akkor lehet, hogy kicsit zavaros a látvány, viszont forgatgatva az objektumot szerintem szépen kiviláglik a két felület helyzete. A köröknek és a gömbnek a tartozékaként megjelenítettem a sugarakat is. Úgy csináltam, hogy a $\vec{\mathbf{P}}$ változtatható felületi pontból a meridián simulókörének színével vonalat húztam $\mathbf{M}\vec{\mathbf{cent}}$-be, majd innen a harántmetszet simulókörének színével vonalat húztam $\mathbf{N}\vec{\mathbf{cent}}$-be. $\mathbf{R}\vec{\mathbf{cent}}$-et pedig egy pont jelöli a két középpont között. Így szépen kivehető a három sugár méreteinek viszonya és az, hogy egy egyenesen vannak. Tehát megjelenik két simulókör, az egyikhez tartozó görbe, és a simulógömb, viszont hol van a másikhoz tartozó görbe, azaz a harántmetszet, amihez a kék simulókör simulna? Ennek a paraméteres egyenletrendszerét nem találtam, és nem is sikerült meghatároznom az adott időn belül. A meghatározás során olyan elvet követtem, ami kevesebb dimenzióban működik, és ezért gondoltam hogy 3D-ben is menni fog. Konkrétan azt szerettem volna, hogy egy síkkal elmetszem az ellipszoidot. Két dimenzióban működik az, hogy két egymást metsző paraméteres módon megadott görbe egyenletrendszerét egyenlővé teszem, és az így kapott egyenletrendszert megoldom. Ekkor megkapom az egyik görbe paraméterének konkrét értékét, ahol a pont van, és ezt visszahelyettesítve a görbe egyenleteibe, megkapom a keresett pont koordinátáit. Két egyenlet, két ismeretlen, egyértelmű megoldás. Gondoltam én, hogyha ezt az elvet követem, és egyenlővé teszem a térbeli felületek egyenletrendszereit, akkor kapok egy új egyenletrendszert már három szimultán egyenlettel, amelyben 4 ismeretlen paraméter

szerepel. Ez alul-határozott. Gondoltam én, hogy ez nagyon jó, mert egy görbét keresek, és az az egy plusz paraméter amely a megoldás után benne lesz a görbe egyenleteiben, az lesz a görbe paramétere. Amilyen logikus az elv (legalábbis szerintem), annyira csúnya lett az eredmény. Mert valóban kijött egy görbe, de az nem a várakozásnak megfelelő volt. Lehet hogy elvi hibát, lehet hogy kivitelezési hibát követtem el. Ezt nem tudom.

Ezeken kívül még berajzoltam a $\vec{\mathbf{P}}$ pontbeli érintőket és három derékszöget, jelezve a sugarak merőlegességét a felületre. Ezek szerkesztése a 2.2.1.-ben leírtakhoz hasonlóan történt, csak itt még plusz egy vektor megjelenik, amihez a másik érintőből és a negatív normálisból is ívet húztam:

$$\vec{\tan}90 := \begin{bmatrix} \cos(\Lambda - 90°) \\ \sin(\Lambda - 90°) \\ 0 \end{bmatrix} + \vec{\mathbf{P}}$$



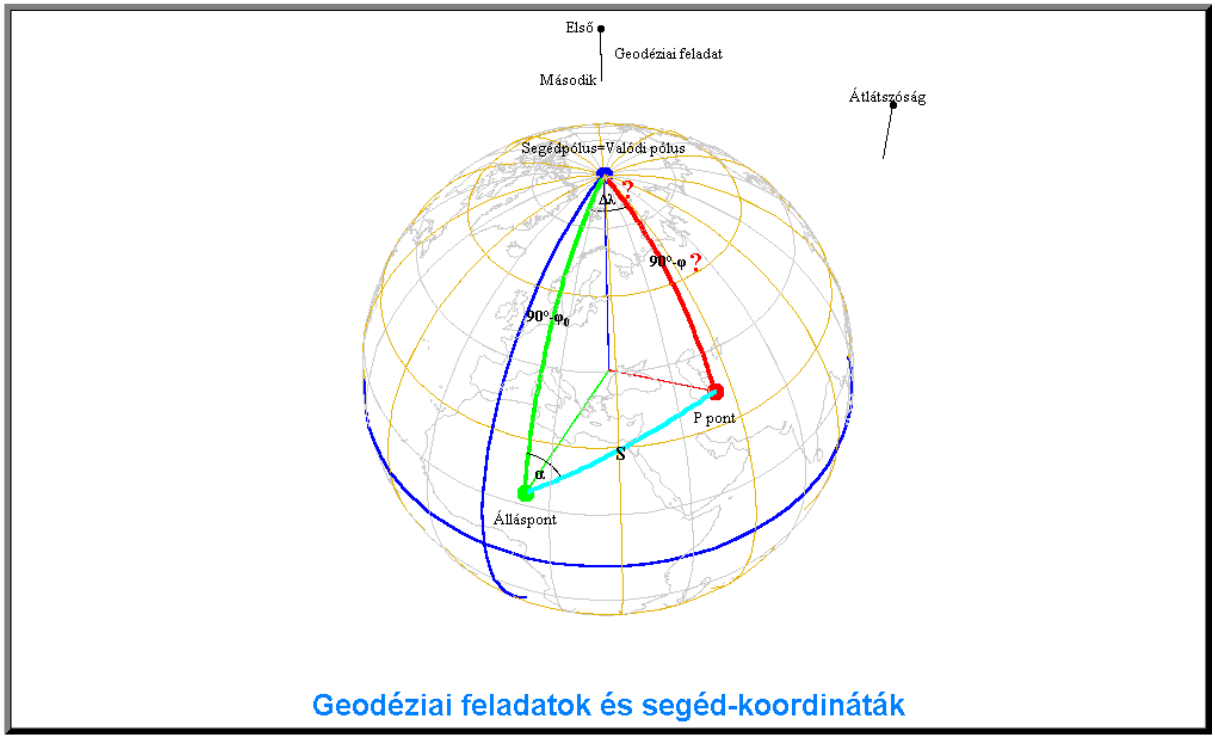
5. ábra

2.2.3. Geodéziai feladatok és segéd-koordináták

Talán ez az egyik leghasznosabb applet mind közül, mivel szeretné közös nevezőre hozni többek között a geodéziai alapfeladatokat, a különböző térinformatikai alkalmazásokat (pl.: városok távolsága) és a térképészetben alapfontosságú segéd-koordináta-rendszert a gömbbel közelített Földön. Ezek mindegyike gömbháromszögtani számításokat igényel, amelyek bemenetei a gömbön mért szögek és távolságok. Az a célom, hogy a felhasználó lássa, hogy az elsőre teljesen különbözőnek látszó számítások alapjában véve ugyanazok, csak a paraméterek között vannak különbségek. Mindegyik számításnál tudjuk egy gömbháromszög két oldalát és közbezárt szögét, és keressük a harmadik oldalát és az egyik maradék szöget. A programban láthatjuk a Föld valódi fokhálózatát és a kontinensek körvonalait szürke színnel, és ezzel együtt a sárga-kék segéd-fokhálózatot is, amely a segédpólus helyzetének függvényében változik. Ezen kívül ortodróma-ívek adják az aktuális gömbháromszög oldalait, melynek egyik végpontja a kérdéses változó pont, amelyet áthelyezhetünk a gömbön. Megírások és körívek segítik az oldalak és a szögek azonosítását, és kapcsolóval választhatjuk ki a szemlélni kívánt feladatot.

Kiemelendő a segéd-koordináta-rendszer fontossága, mert annak koordinátáit használva egységesen kezelhetünk egyes vetületeket, így azok nincsenek kötve kitüntetett helyekhez (pl.: Északi-sark, Egyenlítő), hanem szabadon elforgathatóak a gömbön. A segéd-fokhálózat a

gömb átparaméterezése. Mivel a gyakorlati alkalmazásban általában csak olyan átparaméterezést választunk, ahol a kezdőmeridián tartalmazza az egyik pólust, így csak az ilyen segéd-koordináta-rendszert van módunk a programban beállítani. Viszont választhatunk ennek két esete közül (északi vagy déli pólust tartalmazza). Az ez utáni programokban megjelenő gömbi fokhálózatok segédföldrajzi-fokhálózatoknak tekintendőek.



6. ábra

- **Szerkesztés**

A program készítése közben egyre újabb ötletek és kívánságok merültek fel, ezért sok átírás történt visszamenőleg, és ebből kifolyólag a forráskód sem mindig a leglogikusabban épül föl. Az első applet-hez hasonlóan itt is csináltam átlátszóságot állító kapcsolót. Felületen mozgatható pont is készült, nem is egy, hanem három. Ezek már csak gömbfelületen mozognak, ezért az eddig megszokott **a**-k és **b**-k helyére **r** lép. A pirosat $\vec{\mathbf{P}}=[\mathbf{px},\mathbf{py},\mathbf{pz}]$-vel, a zöldet $\vec{\mathbf{Q}}=[\mathbf{qx},\mathbf{qy},\mathbf{qz}]$-val, a kéket pedig, mint Segédpólust, $\vec{\mathbf{SP}}=[\mathbf{spx},\mathbf{spy},\mathbf{spz}]$-vel jelölöm. $\vec{\mathbf{P}}$ mindig látható, és bárhová mozgatható, $\vec{\mathbf{Q}}$ csak akkor ha a Segédpólus megegyezik a Valódi pólussal. $\vec{\mathbf{SP}}$ mindig látható, de ha a pólus közelébe ér, akkor bepattan a Valódi pólusba. Ezt úgy valósítottam meg, hogy a pont felületen való tartásához szükséges

2.2.1.-ben említett, de nem leírt függvénysor végén történő újbóli értékadás előtt beszúrok egy HA() függvényt, amely megváltoztathatja a koordinátákat:

$$\mathbf{on} := \left[\!\left[\arcsin\left(\frac{\mathbf{spaz}}{\mathbf{r}}\right) > 87° \right]\!\right] \qquad \vec{\mathbf{SP}} := \begin{cases} [10^{-30}\ 0\ \ \mathbf{r}] & \mathbf{ha\ \ on=1} \\ [\mathbf{spx\ \ spy\ \ spz}] & \mathbf{ha\ \ on=0} \end{cases}$$

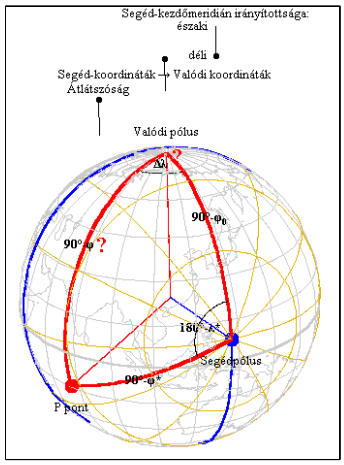
[[..]]-et használtam az Iverson zárójel jelölésére, amely felfogható olyan függvényként, ami az argumentumban lévő feltétel teljesülése esetén 1-et ad eredményül, máskor pedig 0-át. **spaz** pedig $\vec{\mathbf{SP}}$-nek a függvénysoron belüli ideiglenes **Z** koordinátája. Azért választottam 0 helyett a fenti nagyon kicsi pozitív számot, hogy a segédpólus hosszúságának számításakor ne legyen 0-val való osztás és értéke legyen 0. Alapértelmezésben **on=1.** Amikor **on=0**, akkor egy kapcsoló jelenik meg, amellyel a segéd-kezdőmeridián irányítottságát állíthatom. A földi fokhálózat szürke vonalai itt is 15°-ra vannak egymástól, viszont itt egy kicsit bonyolultabb a helyzet, mint eddig. Amikor **on=1**, akkor a 30°-os segéd-fokhálózat egybeesik a valódival, és nem lenne esztétikus két vonalsorozat egymáson. Ezért ha **on=1**, akkor eltűnik a szürke színű 30°-os valódi fokhálózat, melynek komplementere egy statikus, ugyancsak 30°-os, de 15°-kal eltolt hálózat. A kontinenseket 2.2.1.-hez hasonlóan csináltam. A segéd-fokhálózatot sárga és kék színnel jelöltem, és a következőképpen készült: A gömb egyenleteit megszoroztam balról egy $\mathbf{90°-\varphi_{\vec{SP}}}$ szerinti **Y** körüli forgatómátrixszal, majd egy **Z** körüli $\boldsymbol{\lambda_{\vec{SP}}}$ szerintivel (az eredmény nincs leírva, mert túl nagy terjedelmű lenne):

$$\vec{\mathbf{S}}(\mathbf{u},\mathbf{v}) := \mathbf{R_Z}\left(\boldsymbol{\lambda_{\vec{SP}}}\right)\cdot\mathbf{R_Y}\left(\mathbf{90°}-\boldsymbol{\varphi_{\vec{SP}}}\right)\cdot\begin{bmatrix} \mathbf{r}\cdot\cos(\mathbf{u})\cdot\cos(\mathbf{v}) \\ \mathbf{r}\cdot\cos(\mathbf{u})\cdot\sin(\mathbf{v}) \\ \mathbf{r}\cdot\sin(\mathbf{u}) \end{bmatrix} \qquad \begin{aligned} \boldsymbol{\varphi_{\vec{SP}}} &:= \arcsin\left(\frac{\mathbf{spz}}{\mathbf{r}}\right) \\ \boldsymbol{\lambda_{\vec{SP}}} &:= \mathbf{arctan2}(\mathbf{spx},\mathbf{spy}) \end{aligned}$$

A képletet felhasználva a **Table[..]** függvénnyel koordináta-listát állítok elő 30°-onként véve **u**-t és **v**-t, viszont kihagyom **u=0°**-ot**,v=0°**-ot és **180°**-ot a listából. **u=0°**-ot ezzel a módszerrel külön készítem el, mert neki kék színt adok, és ugyanúgy **v=0°**-ot. De ez utóbbit kétszer csinálom meg, sárga és kék színben, mert feltételhez kötöm, hogy melyik jelenjen meg. Ez a feltétel a következő:

$$\mathbf{n} := [[\mathbf{on=0\ \ \&\ \ kapcsoló\ \ állása : északi}]] \qquad \vec{\mathbf{S}}_{\mathbf{v=0°}}(\mathbf{u}) := \begin{cases} \mathbf{sárga} & \mathbf{ha\ \ n=1} \\ \mathbf{kék} & \mathbf{ha\ \ n=0} \end{cases}$$

Sajnos a fordítócsomag nem támogatja az attribútumok külön kezelését, sem az abban szereplő számokat változóval való megadását, ezért a különböző tulajdonságú, de azonos görbéket külön le kell írni. **v=180°** két ortodrómából áll, az egyik $\vec{SP}$ -ből, a másik pedig a déli segédpólusból szerkesztett vonal a valódi pólusba. Az első a gömbháromszög egyik oldala, ez piros színű. A második a színekre nézve felcserélt feltételekkel rendelkezik $\vec{S}_{v=0°}(u)$-hoz képest.



7. ábra

$$\vec{E}_1(g):=\begin{bmatrix} r\cdot\cos\left(g\cdot\varphi_{\vec{SP}}\right)\cdot\cos\left(\lambda_{\vec{SP}}\right) \\ r\cdot\cos\left(g\cdot\varphi_{\vec{SP}}\right)\cdot\sin\left(\lambda_{\vec{SP}}\right) \\ r\cdot\sin\left(g\cdot\varphi_{\vec{SP}}\right) \end{bmatrix};\ \vec{E}_2(g):=\begin{bmatrix} -r\cdot\cos\left(g\cdot\left(180°-\varphi_{\vec{SP}}\right)\right)\cdot\cos\left(\lambda_{\vec{SP}}\right) \\ -r\cdot\cos\left(g\cdot\left(180°-\varphi_{\vec{SP}}\right)\right)\cdot\sin\left(\lambda_{\vec{SP}}\right) \\ r\cdot\sin\left(g\cdot\left(180°-\varphi_{\vec{SP}}\right)\right) \end{bmatrix}\ \text{ahol}\ \ 0\le g\le 1$$

$$\vec{E}_2(g):\begin{cases} \text{sárga} & \text{ha}\ \ n=0 \\ \text{kék} & \text{ha}\ \ n=1 \end{cases}$$

Ezután $\vec{E}_1(g)$ -hez hasonlóan ortodrómát húztam $\vec{P}$ -be (piros) és $\vec{Q}$ -ba (zöld). Megjelenésük feltételei ugyanazok, mint a pontoknak. Létrejött még ezeken kívül két ortodróma. (Utólag visszanézve nagy mértékben egyszerűsödött volna a forráskód, ha $\vec{Q}$ és $\vec{SP}$ koordinátáit összevonva csinálok egy harmadikat, és **on**-tól tettem volna függővé, hogy éppen melyik szolgáltatja az értékét. Ezzel két ortodrómát, megírásokat és egyéb függvényeket megspórolhattam volna, de ez is a forráskód „visszafele épüléséből” adódott.) Egyik ortodróma $\vec{P}$ -ből $\vec{Q}$ -ba megy, ha **on**=1, a másik pedig $\vec{P}$ -ből $\vec{SP}$ -be, ha **on**=0. Ezek már akármilyen ortodrómák lehetnek, nem csak speciálisak, mint az eddigiek. Szerkesztésük bonyolultabb. Először megmértem a két pont távolságának szögértékét:

$da:=\arccos\left(\frac{\vec{P}\cdot\vec{Q}}{r^2}\right)$ Ezután ki kell jelölnöm egy origó középpontú derékszögű koordináta-rendszert, amely együtt forog a két ponttal, mert ebben a rendszerben megadhatom egyszerűen az ortodrómát, mint körívet. Kiválasztom az egyik pont vektorát $\tilde{X}$ tengelynek, legyen mondjuk $\vec{P}$, az $\tilde{Y}$ tengelynek pedig egy erre merőleges vektornak kell lenni a $\vec{P}\,\vec{Q}$

síkban. A vektor: $\left(\vec{\mathbf{P}}\times\vec{\mathbf{Q}}\right)\times\vec{\mathbf{P}}$**.** De ennek a vektornak **r** hosszúságúnak kell lenni, amit úgy érek el, hogy lenormálom, és megnyújtom **r**-szeresére. Így: $\tilde{\mathbf{Y}}:=\frac{\left(\vec{\mathbf{P}}\times\vec{\mathbf{Q}}\right)\times\vec{\mathbf{P}}}{\left|\left(\vec{\mathbf{P}}\times\vec{\mathbf{Q}}\right)\times\vec{\mathbf{P}}\right|}\cdot\mathbf{r}=\begin{bmatrix}\mathbf{oahx}\\\mathbf{oahy}\\\mathbf{oahz}\end{bmatrix}$.

Ezek segítségével a türkizkék ortodróma pedig:

$$\vec{\mathbf{O}}(\mathbf{g}):=\vec{\mathbf{P}}\cdot\cos(\mathbf{g}\cdot\mathbf{da})+\tilde{\mathbf{Y}}\cdot\sin(\mathbf{g}\cdot\mathbf{da})\quad\text{ahol}\quad 0\le\mathbf{g}\le 1$$

A másik ortodrómát is ugyanígy csináltam, csak lecseréltem $\vec{\mathbf{Q}}$-t $\vec{\mathbf{SP}}$-re. Csináltam még két körívet a gömbháromszögben, amelyek szögeket jeleznek. Az első a **Δλ,** amelyet **dl** jelöl a forráskódban. Ezt az értéket -180° és 180° közé kell szorítanom, hogy a dátumválasztón átmérve is helyes értéket kapjak:

$$\begin{matrix}\mathbf{j}:=\lambda_{\vec{\mathbf{P}}}-\mathbf{onl}\ ;\\ \mathbf{k}:=[[\ |\mathbf{j}|>180°\ ]]\ ;\end{matrix}\quad\Delta\lambda:=-\mathbf{sign}(\mathbf{j})\cdot 360°\cdot\mathbf{k}+\mathbf{j}\quad\text{ahol}\quad\mathbf{onl}:=\begin{cases}\lambda_{\vec{\mathbf{SP}}} & \text{ha}\quad\mathbf{on}=1\\ \lambda_{\vec{\mathbf{Q}}} & \text{ha}\quad\mathbf{on}=0\end{cases}$$

Magát a görbét pedig úgy csináltam, hogy az mindig a gömb felületén legyen. Végül is egy parallelkör ívét szerkesztettem meg, úgy hogy **Δλ,** és a pontok pólustól való távolságának függvényében mindig változzon. Vettem a gömbháromszög eme szögét közrefogó két oldal minimumának ötödét, és ezt felhasználtam a görbe létrehozásánál. Ez azért kell, hogy egyaránt tudjunk szemlélődni kis, és nagy méretarányban is. A görbe egyenletrendszere:

$$\vec{\mathbf{DL}}(\mathbf{g}):=\mathbf{R_Z}(\mathbf{onl})\cdot\begin{bmatrix}\mathbf{r}\cdot\sin(\mathbf{dlm})\cdot\cos(\Delta\lambda\cdot\mathbf{g})\\ \mathbf{r}\cdot\sin(\mathbf{dlm})\cdot\sin(\Delta\lambda\cdot\mathbf{g})\\ \mathbf{r}\cdot\cos(\mathbf{dlm})\end{bmatrix}$$

$$\text{ahol}\quad\mathbf{dlm}:=\frac{\min\left(90°-\varphi_{\vec{\mathbf{P}}},90°-\mathbf{onf}\right)}{5};\quad\mathbf{onf}:=\begin{cases}\varphi_{\vec{\mathbf{SP}}} & \text{ha}\quad\mathbf{on}=1\\ \varphi_{\vec{\mathbf{Q}}} & \text{ha}\quad\mathbf{on}=0\end{cases}\quad\text{és}\quad 0\le\mathbf{g}\le 1$$

Ehhez hasonlóan a második szög-ívet is megcsináltam apró módosításokkal (8. és 9. ábra), de előtte szót ejtek a szög (**ona**) kiszámításáról. Ez a szög ha **on=1**, az azimut, ha **on=0**, a segédhosszúság. Számításuknál a második (inverz) geodéziai feladat második lépését kell végrehajtanom. Itt előbb a szög koszinuszát számítottam ki, majd **Δλ** előjelétől tettem függővé, hogy melyik a helyes a két lehetséges érték közül:

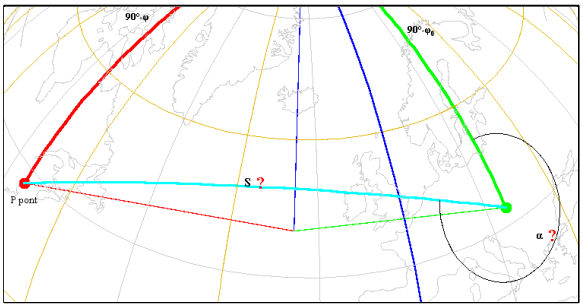
$$\mathbf{aa}:=\arccos\left(\frac{\sin(\varphi_{\vec{\mathbf{SP}}})-\sin(\mathbf{onf})\cdot\cos(\mathbf{ond})}{\cos(\mathbf{onf})\cdot\sin(\mathbf{ond})}\right);\qquad\mathbf{a}:=\begin{cases}360°-\mathbf{aa} & \text{ha}\quad\Delta\lambda<0\\ \mathbf{aa} & \text{ha}\quad\Delta\lambda\ge 0\end{cases};\qquad\mathbf{sl}:=\mathbf{sign}(\Delta\lambda)\cdot\mathbf{aa}\ ;$$

$$\rightarrow\qquad\underline{\mathbf{ona}}:=\begin{cases}\mathbf{a} & \text{ha}\quad\mathbf{on}=1\\ \mathbf{sl} & \text{ha}\quad\mathbf{on}=0\end{cases};\qquad\text{ahol}\quad\mathbf{ond}:=\begin{cases}\mathbf{da} & \text{ha}\quad\mathbf{on}=1\\ \mathbf{dc} & \text{ha}\quad\mathbf{on}=0\end{cases}$$

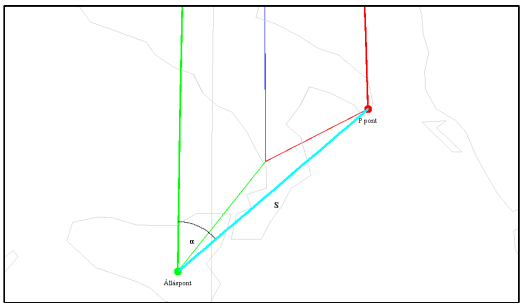
Miután kiszámoltam a szöget, megadom a görbe egyenletrendszerét, amely alapján a koordinátalistát a szokásos módon megalkotom a **Table[..]** függvény segítségével a *Mathematica*-ban. Hasonlóan $\vec{\mathbf{DL}}(\mathbf{g})$-hez a görbe ilyen alakot vesz fel, csak itt belép még egy forgatás:

$$\vec{\mathbf{A}}(\mathbf{g}) := \mathbf{R_Z}(\mathbf{onl})\cdot\mathbf{R_Y}(90°-\mathbf{onf})\cdot\begin{bmatrix}-\mathbf{r}\cdot\sin(\mathbf{onm})\cdot\cos(\mathbf{ona}\cdot\mathbf{g})\\-\mathbf{r}\cdot\sin(\mathbf{onm})\cdot\sin(\mathbf{ona}\cdot\mathbf{g})\\\mathbf{r}\cdot\cos(\mathbf{onm})\end{bmatrix}$$

$$\text{ahol}\quad \mathbf{onm} := \frac{\min(\mathbf{ond}, 90°-\mathbf{onf})}{5};\quad \text{és}\quad 0\le\mathbf{g}\le 1$$



8. ábra



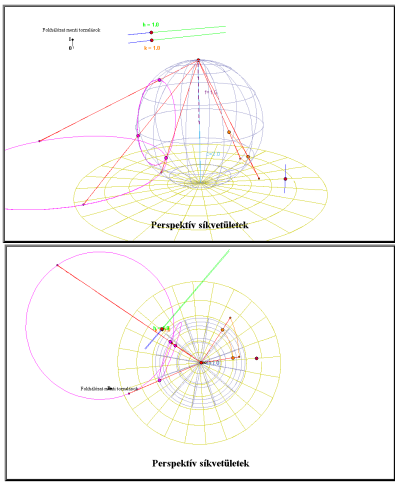
9. ábra

Ezek után már csak a megírások maradtak hátra. Minden szöveg a kapcsolódó görbék és pontok koordinátáit használja, ezáltal együtt mozogva az objektummal. A megírások közül kiemelném a piros kérdőjeleket, mert ezek fontos szerepet vállalnak az applet céljának elérésében. A fentebb említett feladatok céljaihoz tettem a kérdőjeleket, jelezve, hogy az adott helyzetben mit kell kiszámolni. A gömbháromszög öt használt adatából három ismert, a maradék mellett pedig kérdőjel szerepel. Például az első geodéziai feladatnál ismertek az álláspont koordinátái, a keresett pont távolsága és azimutja, és keressük a P pont koordinátáit, ahol a kérdőjelek vannak. Ezeket a kérdőjeleket a valódi pólus fölötti kapcsolóval állíthatjuk, amelynek megírása is mindig az aktuális helyzetet mutatja. A megírásoknál állíthatjuk a szöveg térbeli helyét, és az attól a képernyő síkjában való eltolását is. Az applet címét cselesen egy origóba helyezett ponthoz rendeltem hozzá, majd a szöveget a képernyő síkjában eltoltam az applet ablakának felső részére.

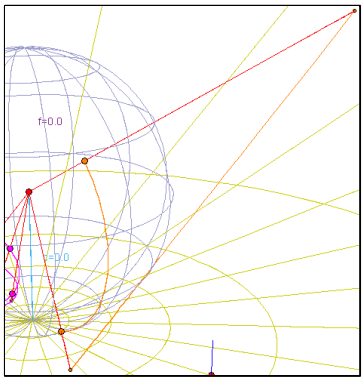
2.2.4. Perspektív síkvetületek

Egy nagyon fontos vetületcsoportot mutat be ez a program. A perspektív vetületek talán a legrégebbi vetületek. Maga a vetület/vetítés szavak is innen származnak, és lettek kiterjesztve

nem geometriai levezetésű társaikra. A vetületek rendszerezése a segédfokhálózat alakján (pl.: valódi kúpvetület), és azon belül annak torzulásain (pl.: szögtartó) alapul. A segédfokhálózat alakja szerinti besorolás a perspektív leképezés során létrejött fokhálózat tulajdonságait örökölte. Én ezek közül a kúpvetületek egy speciális esetét, a síkvetületeket jelenítem meg programommal. Az itt megjelenő fokhálózat az előző applet-ben bemutatott segédfokhálózatot jelenti, tehát az itt megjelenített elemek szabadon elforgathatóak a gömbbel közelített Föld középpontja körül. Lehetősége van a felhasználónak minden vetületi paramétert bizonyos határok közt állítani, így szerintem egy igen látványos, élvezhető és hasznos applet jött létre. Beállíthatjuk a három legtöbbet használt esetet: gnomonikus vetület, ahol az alapfelületi geodéziai vonal vetületi képe ugyancsak geodéziai vonal; sztereografikus vetület, amely szögtartó, így országunkban is használták geodéziai felmérésekhez; ortografikus vetület, amely a Földet ahhoz képest nagyon messziről nézve reprezentálja. Természetesen a köztes helyzeteket is szemlélhetjük. Látható két alapfelületi görbe: egy kiskör és egy nagykör részlete, amelyeknek vetületi képeit vizsgálhatjuk az adott beállítások mellett. A síkvetületek esetében a szögtartó eset egyben körtartó is, ezért ilyenkor akárhogyan állítjuk be a kiskört, az a képfelületen is kör lesz (10. ábra). Ezt most használják például földrengések tenzorainak reprezentálásánál is, régebben pedig a vetület szerkesztéséhez használták. A gnomonikus esetben pedig a nagykör részlete, azaz egy ortodróma képe lesz mindig egyenes a képfelületen (11. ábra).



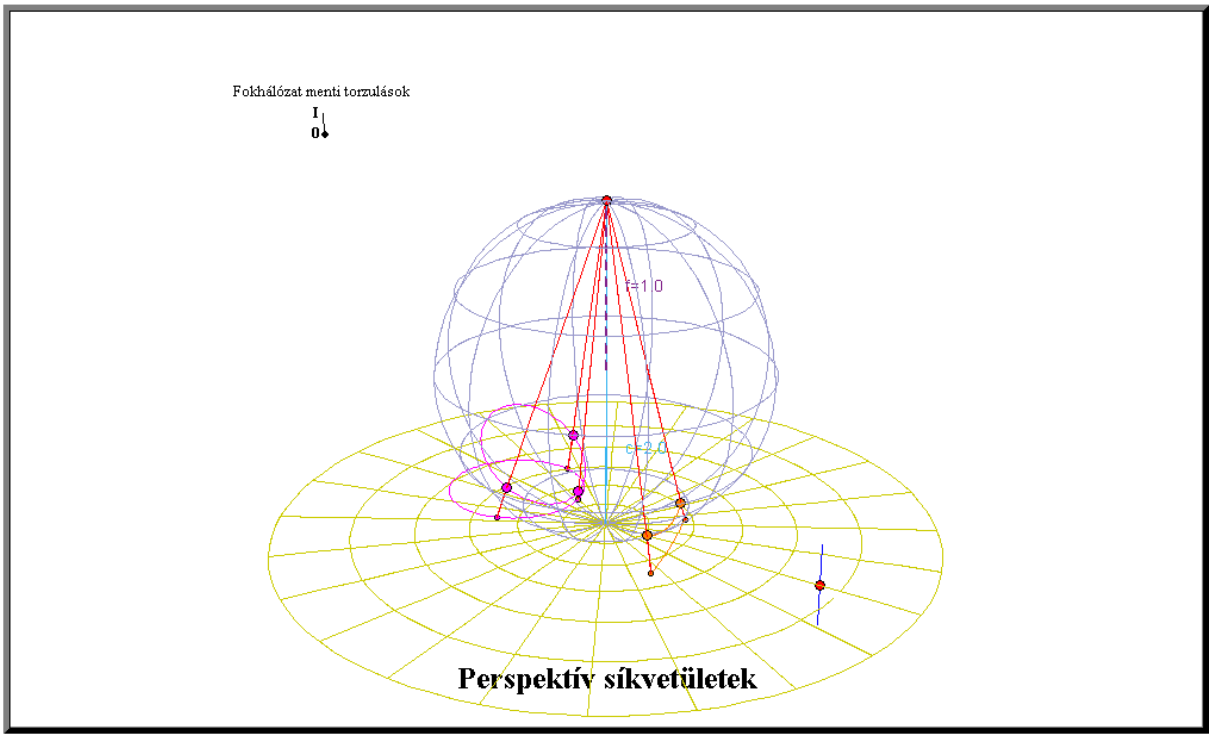
10. ábra



11. ábra

Ezek mellett még megjeleníthetőek a programban a torzulások. A felhasználó beállíthatja a meridián-, és a parallelkör menti torzulás értékét, és ekkor megjelenik az ehhez rendelhető alap-, és képfelületi pontok halmaza a megfelelő színezéssel. A torzulásokat csak bizonyos korlátok között állíthatjuk. Ezek a korlátok erősen függenek a többi aktuálisan beállított paramétertől. Például ortografikus esetben a h parallelkörmenti torzulást nem is mozdíthatjuk ki 1-ből, mivel a vetület abban hossztartó, a meridiánmenti torzulás pedig 0 és 1 között állítható. Viszont sztereografikus

esetben mindkét torzulásnál az alsó határ kis mértékben változik, de felső határt 5-nél szabtam meg a végtelen helyett. A kétrétegű leképezéseket úgy zártam ki, hogy a torzulásokat nem engedem negatív irányba állítani.



12. ábra

- **Szerkesztés**

Ennek a programnak a forráskódja talán a legösszeszedetlenebb, mi sem bizonyítja jobban, hogy néhol beakad, tehát 0-val való osztás, vagy komplex gyök keletkezik. Ez részben attól is van, mert az alkalmazott függvények néhol elég bonyolultak, és eset-szétválasztáskor valószínűleg rossz értéket adtam meg valahol, másrészt időben ez az egyik legelső applet, ahol a fordítócsomagot és magát a programozást is még csak kóstolgattam.

Az előző applet-ekhez hasonlóan itt is szerkesztek egy alapfelületi drótvázat, és öt mozgatható felületi pontot. Ebből kettő egy ortodrómáért felel, és ezeket nem engedem a segédegyenlítőn túllépni, mert a vetületi segédfokhálózat is csak az egyik félgömbhöz jelenik meg. Az ortodrómát az előző programban leírtakhoz hasonlóan készítettem el. A másik három pont az általános gömbfelületi kiskört határozza meg egyértelműen. Ennek először megkerestem a középpontját. Ezt úgy csináltam, hogy először is a három pont által meghatározott síkon

megkerestem a háromszög egyik oldalfelező egyenesének egyenletét:

$$\vec{A}(u) := \left(\left(\left(\vec{Q}-\vec{P}\right)\times\left(\vec{R}-\vec{P}\right)\right)\times\left(\vec{Q}-\vec{P}\right)\right)\cdot u \; + \; \frac{\vec{Q}+\vec{P}}{2}$$

Majd ugyanígy egy másikét, ahol $\vec{Q}$ -t és $\vec{R}$ -et felcseréltem. Az így kapott két egyenes metszéspontját, tehát a kiskör középpontját 2.2.2. vége felé bemutatott módszer segítségével határoztam meg $(\vec{Kcent})$. Ez egyébként a gömb belsejében helyezkedik el. Majd ezután a középpont és az egyik pontnak a távolsága adta a kör sugarát (**kr**). Ezután megfogalmaztam **XY** síkban egy kört, elforgattam a kör középpontjának koordinátáival, és eltoltam $\vec{Kcent}$-be. Íme:

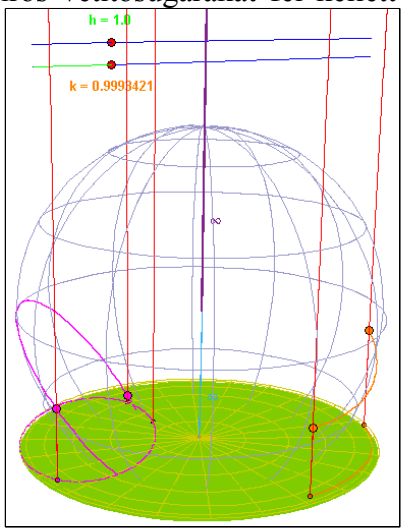
$$\vec{K}(g) := R_Z\left(\lambda_{\vec{Kcent}}\right)\cdot R_Y\left(90° - \varphi_{\vec{Kcent}}\right)\cdot\begin{bmatrix} kr\cdot\cos(g) \\ kr\cdot\sin(g) \\ 0 \end{bmatrix} + \vec{Kcent}$$

ahol

$$\varphi_{\vec{Kcent}} := \arctan\left(\frac{Kcentz}{\sqrt{Kcentx^2 + Kcenty^2}}\right); \qquad \lambda_{\vec{Kcent}} := \operatorname{arctan2}(Kcentx, Kcenty)$$

Ezt máshogyan is megszerkeszthettem volna, de örültem, hogy működik. Az elején még nem is működött, nem is értettem, hogy miért, de egy fizikus fórumon leírtaktól megvilágosodtam, és megnöveltem a gömb sugarát 400-szorosára. De még így is, ha a kör középpontját a gömbéhez közel viszem, akkor megbolondul.

A vetítési középpontot egy kontrollponttal mozgathatjuk. Ha az a gömb belsejében van, akkor a két pont megegyezik, viszont azon kívül már egy jól illesztett exponenciális függvény szerint távolodnak, hogy a felhasználó kényelmesen tudja a távolba helyezni a vetítési középpontot. Ebből a pontból az öt felületi pontba húzódó piros vetítősugarakat fel kellett darabolni, mégpedig azért mert ortografikus módban a vonalak könnyen metszhették volna a képernyő síkját, és eltűntek volna.

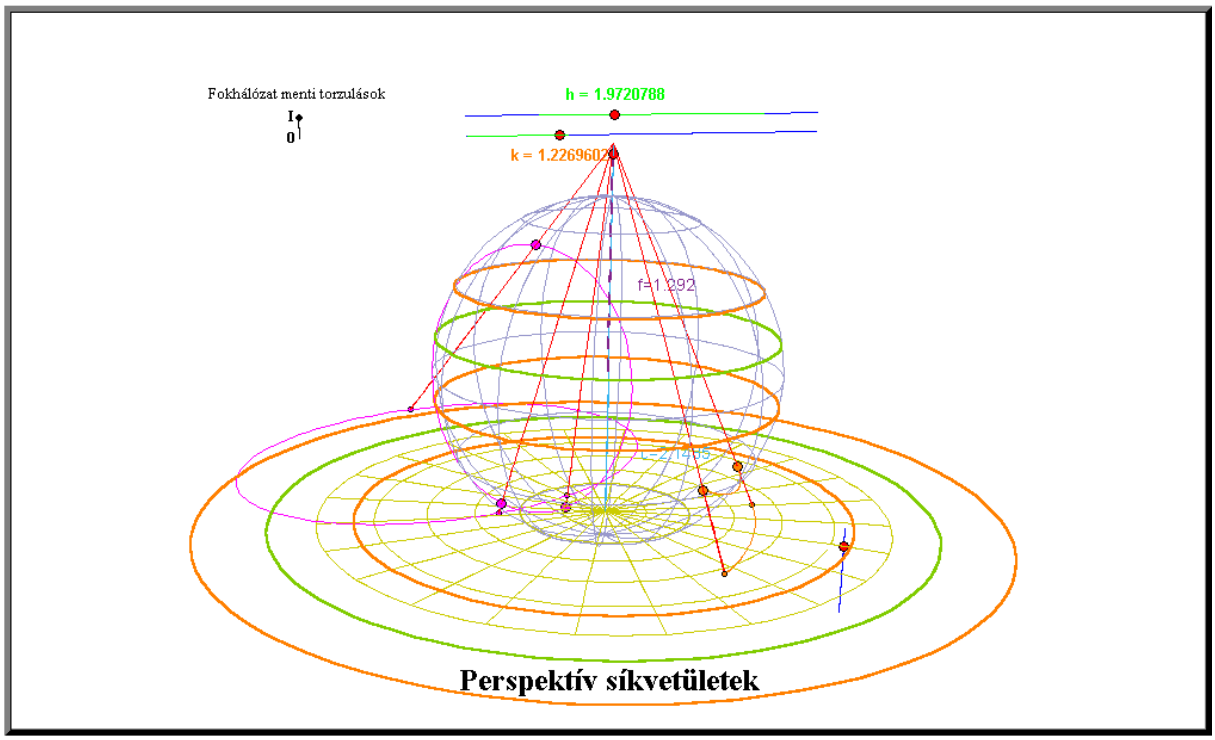


13. ábra

A már említett torzulások korlátainak számítását, és az azokon belül változtatható értékekből a földrajzi helyekre való visszaszámítást nem írom le, mert amúgy még ennél is szárazabb lenne a dolgozat, és eléggé elkanyarodna az eredeti témától. Egyébként ezek a számítások differenciálszámításról, és komplex gyökök elkerüléséről szólnának. A torzulások

megjelenítésénél kiemelném az ortografikus vetületet, ahol megjelenik egy zöld tárcsa, amely azt hivatott jelezni, hogy a parallelkör menti hossztorzulás mindenhol **1** (13. ábra).

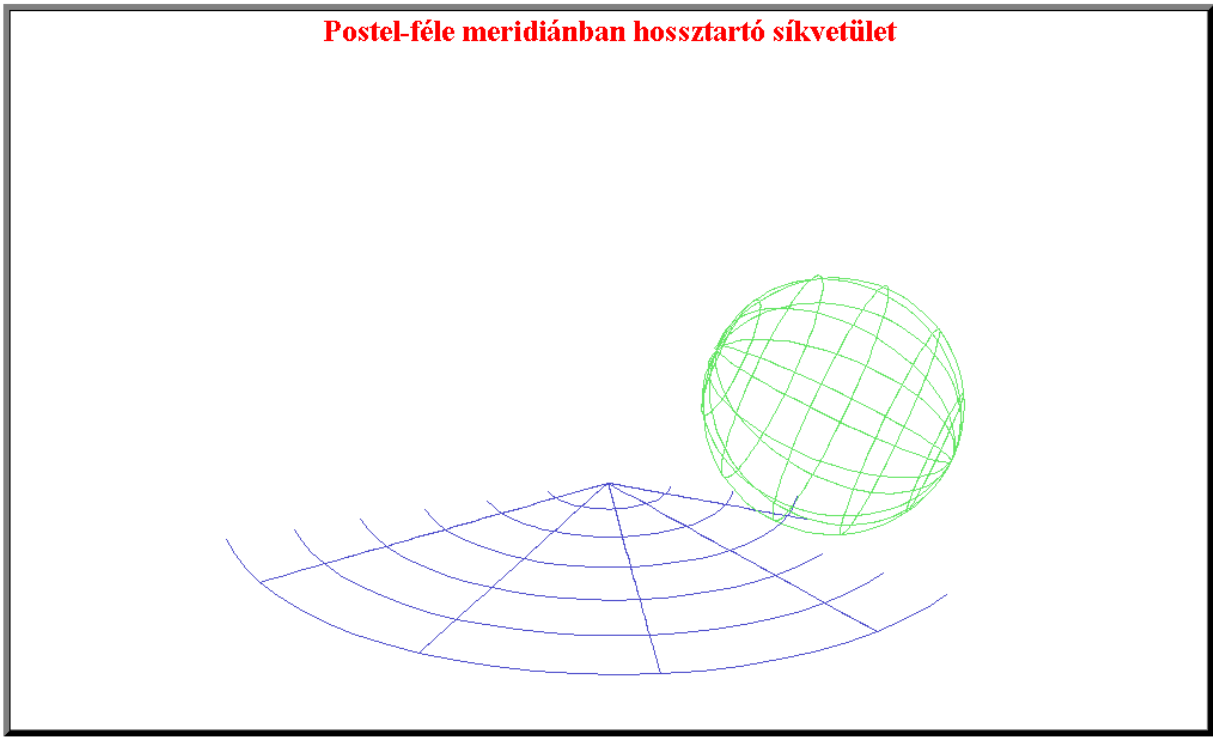
A képfelületi objektumokat összefoglalom annyiban, hogy az alapfelületi görbék, és a vetítési középpont által meghatározott felületek paraméteres egyenletrendszereit egyenlővé tettem a képfelületéével, és így megkaptam az adott görbe vetületi képét. A vetületi egyenletek tehát a leképezésnél nem játszottak szerepet.



14. ábra

2.2.5. Postel-féle síkvetület

A vetületek egyik lehetséges és logikusan felépített csoportosítása a vetületi torzulásokon alapul, és ezekből vezeti le a vetületeket. Ez, és az ez utáni két applet ettől elrugaszkodik, és a bemutatott vetületeknek olyan tulajdonságait mutatja be, amelyeknek geometriai töltetük van. Ez a program a meridiánban hossztartó valódi síkvetületet mutatja be. Az applet alapötlete az, hogy a vetület úgy készül, hogy a gömb alapfelületet meridiánjain elgördítjük, és az lenyomatot hagy maga után a síkon. Ezzel egy elég látványos és megjegyezhető vázlatot kap a felhasználó. Ilyen gondolatmenettel a vetület végtelen kicsiny hosszúságváltozással és végtelen sok idő alatt képezné le az alapfelületet, ezért ezt a változást 30°-ra állítottam be, tehát **Δλ=30°**-onként képeződik le egy meridián, és a parallelkörök íveinek környező része.



15. ábra

- **Szerkesztés**

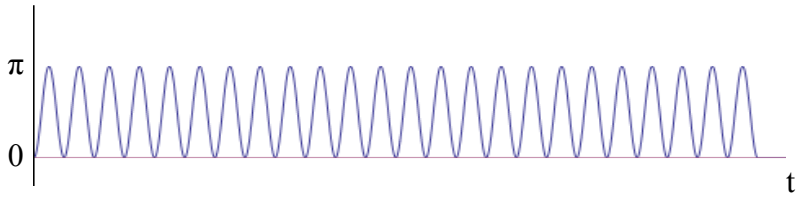
Ez az első animáció szakdolgozatomban. Annyiban különbözik az eddigiektől, hogy a független paraméter most nem az egér mozgásától, hanem a ciklus megállás nélküli ismétlődésétől függ. A mozgó gömböt reprezentáló görbesereget egy kicsi csalás segítségével szerkesztettem meg. Az látható az animációban, hogy minden meridián leképeződik egy külön vonallá, de ehelyett mindig csak a kezdőmeridiánon gurul el a gömb. Ez nem helyes, viszont nem látszik, mert ugyanolyan az összes meridián. Ha megváltoztatnám az egyik meridián színét, látni lehetne, hogy mindig ugyanazon a meridiánon gurul el a gömb. Ezzel megspóroltam egy forgatómátrixot, amely az „előforgatásért" lenne felelős:

$$\vec{\mathbf{F}}(\mathbf{u},\mathbf{v}) := \mathbf{R_Z}(\mathbf{b})\cdot\mathbf{R_Y}(\mathbf{a})\cdot\cancel{\mathbf{R_Z}(-\mathbf{b})}\cdot\begin{bmatrix}\mathbf{r}\cdot\cos(\mathbf{u})\cdot\cos(\mathbf{v})\\ \mathbf{r}\cdot\cos(\mathbf{u})\cdot\sin(\mathbf{v})\\ \mathbf{r}\cdot\sin(\mathbf{u})\end{bmatrix}+\begin{bmatrix}\mathbf{r}\cdot\mathbf{a}\cdot\cos(\mathbf{b})\\ \mathbf{r}\cdot\mathbf{a}\cdot\sin(\mathbf{b})\\ \mathbf{0}\end{bmatrix}$$

A gömb forgatásáért és eltolásáért felelős értékek (**a** és **b**) az időparaméternek a függvényei. Mivel ez egy meridiánban hossztartó vetület, ezért a pólustávolság (**a**) megegyezik a leképezett meridiánnal, ezáltal a középpont által megtett úttal is. Így használhatom **a**-t a vízszintes tengelyű elforgatáshoz, és az eltoláshoz is. A gömb gurulásának esztétikusnak kell

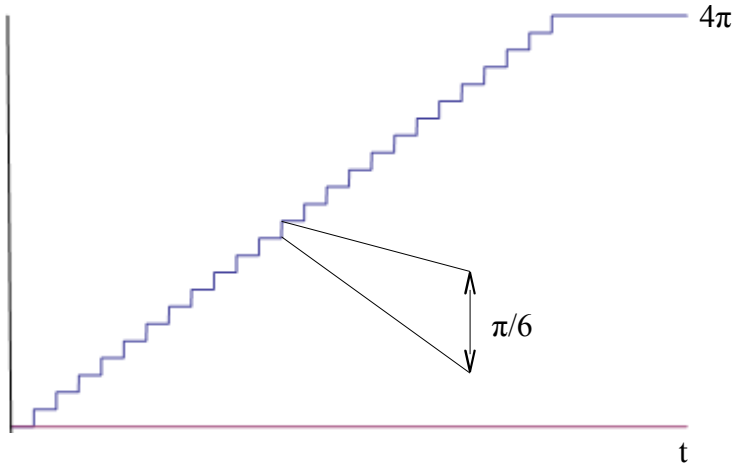
lennie, ezért az **a** paraméter értékét egy sima periodikus időfüggvény adja:

$$\mathbf{a} := \frac{\mathbf{1}-\cos(\mathbf{t})}{\mathbf{2}}\cdot\pi = \boldsymbol{\beta} \quad \rightarrow$$



A másik paraméter függvénye viszont már nem sima. Ez adja az aktuálisan leképeződő meridián szélességét, tehát hogy merre gurul éppen a gömb. Ez viszont már nem változhat folytonosan, így a függvény:

$$\mathbf{b} := \mathbf{floor}\left(\frac{\mathbf{t}}{\mathbf{2}\cdot\pi}\right)\cdot\frac{\pi}{\mathbf{6}} = \boldsymbol{\lambda} \quad \rightarrow$$



Ezzel készen is lett az alapfelület. Következzen a képfelület:

Így utólag visszanézve ezt egyszerűbben is megcsinálhattam volna, de az eredmény ugyanaz. A képsíkon lévő vetületet kis vonalak alkotják, és ha teljesül a hozzájuk rendelt feltétel, akkor megjelennek. Ez a feltétel a következő:

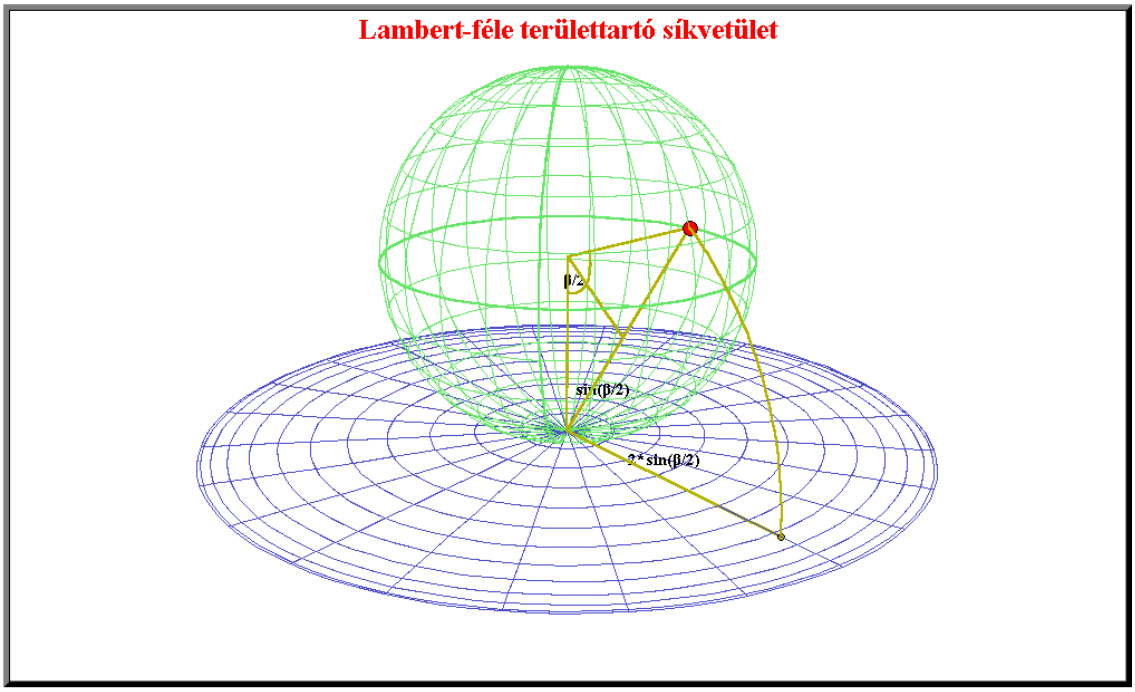
$$\left[(\mathbf{a}\geq \mathrm{B})\,\&\left(\mathbf{b} > \Lambda - \frac{\pi}{\mathbf{12}}\right)\right] \quad | \quad \left[\frac{\mathbf{t}}{\mathbf{12}} > \Lambda + \frac{\pi}{\mathbf{12}}\right]$$ ahol **B** és **Λ** valamely vonal egyik koordinátája.

A vonalakat egy háromszintű **Table[..]** parancs segítségével szerkesztettem meg, felhasználva a fenti feltételt.

Az applet egyik paramétere (**SPIN_X**) a nézőpont keringésének kezdeti értéke. Ezt az animáció közben felülbírálhatjuk, és mozgathatjuk a nézőpontot a megszokott módon. A címet a **PlotLabel[..]** paranccsal jelenítettem meg.

2.2.6. Lambert-féle területtartó síkvetület

A bemutatandó gömb alapfelületű vetületet a valódi síkvetületek közé soroljuk, és ezeken belül ez az egyetlen területtartó. A vetület meglepő módon Lambert nevéhez fűződik. A torzulási viszonyokból kiinduló levezetés a területtartás alapegyenletével kezdődik és eredményül kapjuk a sugárfüggvényt. Viszont ez a sugárfüggvény megegyezik a vetületi kezdőpont és az alapfelületi pont térbeli távolságával, ha a két felületet úgy helyezzük el érintőlegesen, hogy a segédpólus és annak vetületi képe térben egy pontba essen. Míg az előző vetületnél a felületi távolság, itt a térbeli távolság egyenlő a vetületi távolsággal a vetületi középpontból mérve. Az applet elve, hogy egy pont által befolyásolt térbeli háromszög oldalai és szögei segítségével beláttasson a felhasználóval egy másik levezetési módszert. A segédpólus, az origó és valamely felületi pont alkotnak egy egyenlő szárú háromszöget, melynek csúcsánál keletkező szög a pólustávolság (**:=β**). A háromszöget szimmetriatengelyével elfelezve kapunk két egybevágó derékszögű háromszöget, melyeknek egy-egy szöge **β/2**, és átfogója a gömb sugara (térképészetben általában **r=1**). Ezen szöggel szemben lévő oldal hossza $\mathbf{r}\cdot\mathbf{sin}(\boldsymbol{\beta}/2)$, ami pont fele az egyenlő szárú háromszög alapjának, így az alap egyenlő $\mathbf{2}\cdot\mathbf{r}\cdot\mathbf{sin}(\boldsymbol{\beta}/2)$ -vel. Ez megegyezik az elsődleges levezetés eredményével, a sugárfüggvénnyel. Ezt lehet úgy is értelmezni, hogy az alapfelületi pont egy segédpólus középpontú körön keresztül levetül a vetületi síkra (16. ábra).



16. ábra

Nem árt hangsúlyozni, hogy ennek a fontossága sokad-rangú. A vetületet területtartósága miatt használjuk, és nem tulajdonítunk a fentebb leírt dolgoknak nagy jelentőséget. Hasznát abban látom a programnak, hogy aki hirtelen nem emlékszik a képletre és nem tudja levezetni a $\mathbf{h}\cdot\mathbf{k}=\mathbf{1}$ diffegyenletet, az ezzel az egyszerű gondolatmenettel megoldja a problémát.

- **Szerkesztés**

A gömböt előzőekhez hasonlóan csináltam. A fokhálózat vetületi képét pedig:

$$\vec{\mathbf{V}}(\mathbf{u},\mathbf{v}):=\begin{bmatrix}2\,\mathbf{r}\sin(\mathbf{u}/2)\cdot\cos(\mathbf{v})\\2\,\mathbf{r}\sin(\mathbf{u}/2)\cdot\sin(\mathbf{v})\\-\mathbf{r}\end{bmatrix}\quad\textbf{ahol}\quad\begin{matrix}-90°\le\mathbf{u}\le 90°\\-180°<\mathbf{v}\le 180°\end{matrix}$$

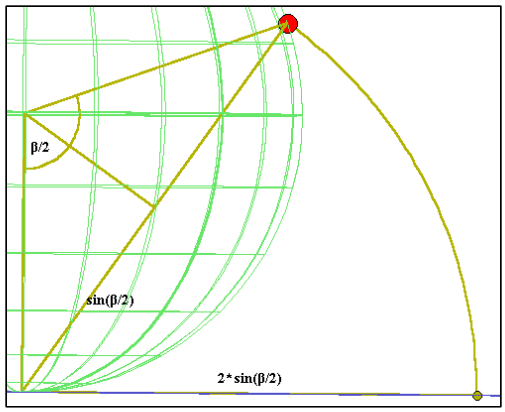
Létrehoztam egy felületi pontot: $\vec{\mathbf{P}}=[\mathbf{x},\mathbf{y},\mathbf{z}]$. Kiszámoltam $\vec{\mathbf{P}}$ pont és a vetületi kezdőpont távolságát $\mathbf{d}:=\sqrt{\mathbf{x}^2+\mathbf{y}^2+(\mathbf{z}+\mathbf{r})^2}$, majd ilyen sugárral körívet húzok a felületi- és a vetületi pont közé:

$$\vec{\mathbf{G}}(\mathbf{g}):=\begin{bmatrix}\mathbf{d}\cdot\cos(\mathbf{g}\cdot\boldsymbol{\beta}/2)\cdot\cos(\boldsymbol{\lambda})\\\mathbf{d}\cdot\cos(\mathbf{g}\cdot\boldsymbol{\beta}/2)\cdot\sin(\boldsymbol{\lambda})\\\mathbf{d}\cdot\sin(\mathbf{g}\cdot\boldsymbol{\beta}/2)\end{bmatrix}\quad\textbf{ahol}\quad\begin{matrix}\boldsymbol{\beta}:=\arcsin\left(-\frac{\mathbf{z}}{\mathbf{r}}\right)\\0\le\mathbf{g}\le 1\end{matrix}$$

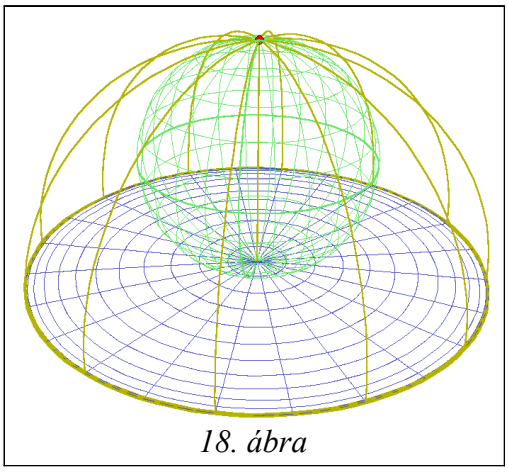
A következő körív a gömb középpontjánál van, és $\boldsymbol{\beta}$-t kívánja megjelölni:

$$\vec{\mathbf{B}}(\mathbf{g}):=\begin{bmatrix}\mathbf{r}/5\cdot\sin(\mathbf{g}\cdot\boldsymbol{\beta})\cdot\cos(\boldsymbol{\lambda})\\\mathbf{r}/5\cdot\sin(\mathbf{g}\cdot\boldsymbol{\beta})\cdot\sin(\boldsymbol{\lambda})\\-\mathbf{r}/5\cdot\cos(\mathbf{g}\cdot\boldsymbol{\beta})\end{bmatrix}\quad\textbf{ahol}\quad 0\le\mathbf{g}\le 1$$

Meghúztam a háromszögek és a sugárfüggvény vonalait is, majd elhelyeztem szövegeket a megfelelő oldalakhoz és szögekhez (17. ábra).



17. ábra

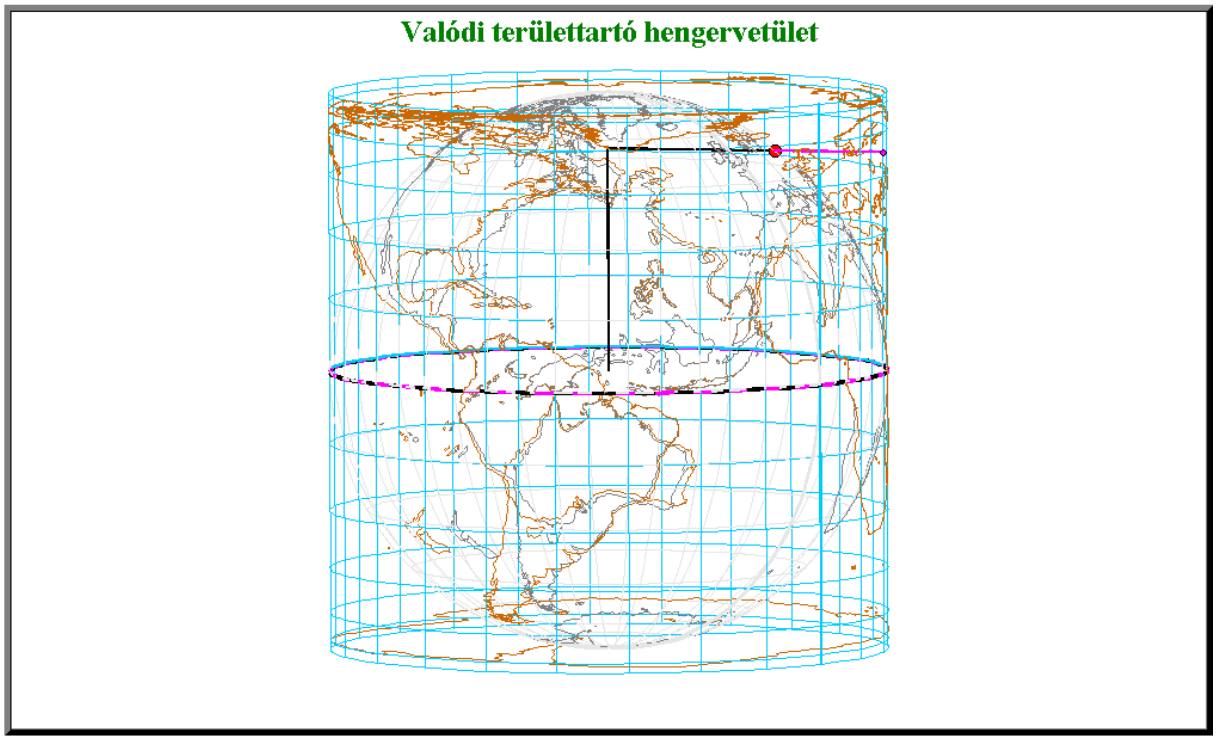


18. ábra

A vetület pólusvonalas, ezért a felületi pontot a 2.2.3.-ben lévő $\vec{\mathbf{SP}}$-hez hasonlóan alkottam meg. Így ha a pont a másik segédpólusba ér, akkor a fentebb leírtak eltűnnek, és megjelenik a vetület kontúrvonala, mint a pólus vetülete és az azt a segédpólussal összekötő tizenkét 2***r** sugarú negyed-körív (18. ábra).

2.2.7. Valódi területtartó hengervetület

Ez a program bemutatja a valódi területtartó hengervetületek csoportját. Az utóbbi két vetületnek csak egy-egy lehetséges változata volt, ennél viszont végtelen sok. Ezt állíthatjuk is a programban egy fogantyúval, amely a valódi hengervetületek általános képletében szereplő c változó értékét állítja. Ez az érték az esetek egy részében a normálparallelkör sugarával egyezik meg. A program alapötlete, hogy a végtelen lehetőség közül egynek adhatunk geometriai jelentést, mégpedig egyenlítőben hossztartó változatnak. Itt mivel **c=1**, ezért **y=sin(φ)**, amelyet úgy is értelmezhetünk, hogy egy $\vec{\mathbf{P}}$ gömbfelületi pontnak és a normális elhelyezésű hengerfelületi képének a **Z** koordinátái egyenlőek. Emiatt a tulajdonság miatt szokták néha ezt a vetületet alkalmazni igényes vizualizációknál (pl.:híradó kezdése). Ha a programban a fogantyú által **c=1**, akkor megjelenik a mozgatható $\vec{\mathbf{P}}$ pont és annak képe összekötve egy egyenessel. A többi esetben nincsen geometriai kapcsolat, viszont lehetősége van a felhasználónak szemlélni azt, hogy hogyan változik a képfelület paramétervonalainak aránya, miközben területtartása, ezáltal a felületének nagysága változatlan. Kiemeltem két esetet: Az egyenlítőben hossztartó vetületet, amely Lambert™-ről kapta nevét, és a **φ=30°**-ban hossztartó vetületet, amely Behrmann-ról kapta a nevét. A program az előző vetületektől eltérően megjeleníti a kontinenseket, mert ez általában csak normális



19. ábra

elhelyezésben használatos.

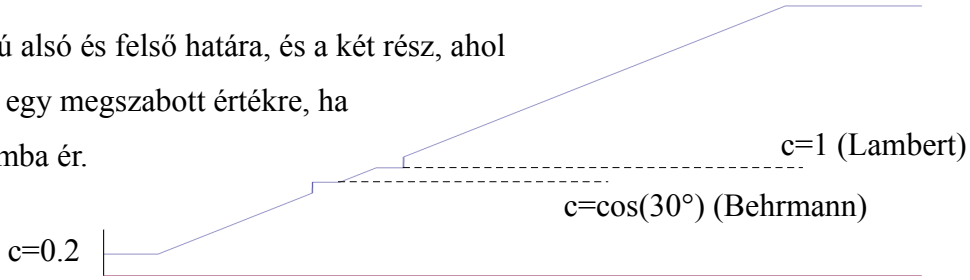
- **Szerkesztés**

Az alapfelületet és az azon lévő kontinensek kontúrvonalait az eddig leírtak alapján alkottam meg. A képfelületet és az azon lévő kontinensek kontúrvonalait pedig a vetületi egyenletek segítségével csináltam:

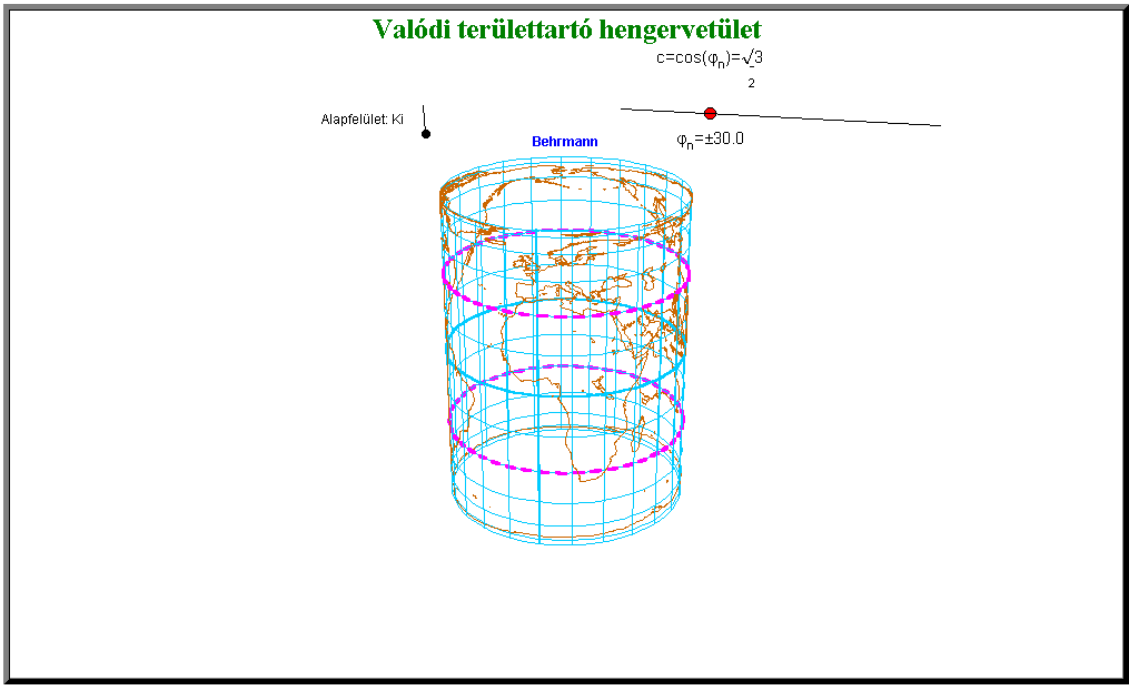
$$\vec{K}(u,v) := \begin{bmatrix} c\cdot\cos(v) \\ c\cdot\sin(v) \\ \frac{\sin(u)}{c} \end{bmatrix} \quad \text{ahol} \quad \begin{matrix} -90° \le (u = \varphi) \le 90° \\ -180° < (v = \lambda) \le 180° \end{matrix}$$

Az alapfelületet egy kapcsolóval kikapcsolhatóvá tettem az átláthatóság kedvéért. Az említett **c** paramétert a programban **r3** reprezentálja. Ezt változtathatjuk, és ennek a felhasználótól függő értéknek a grafikonja a következő:

Itt látható a fogantyú alsó és felső határa, és a két rész, ahol a fogantyú bepattan egy megszabott értékre, ha egy adott intervallumba ér.



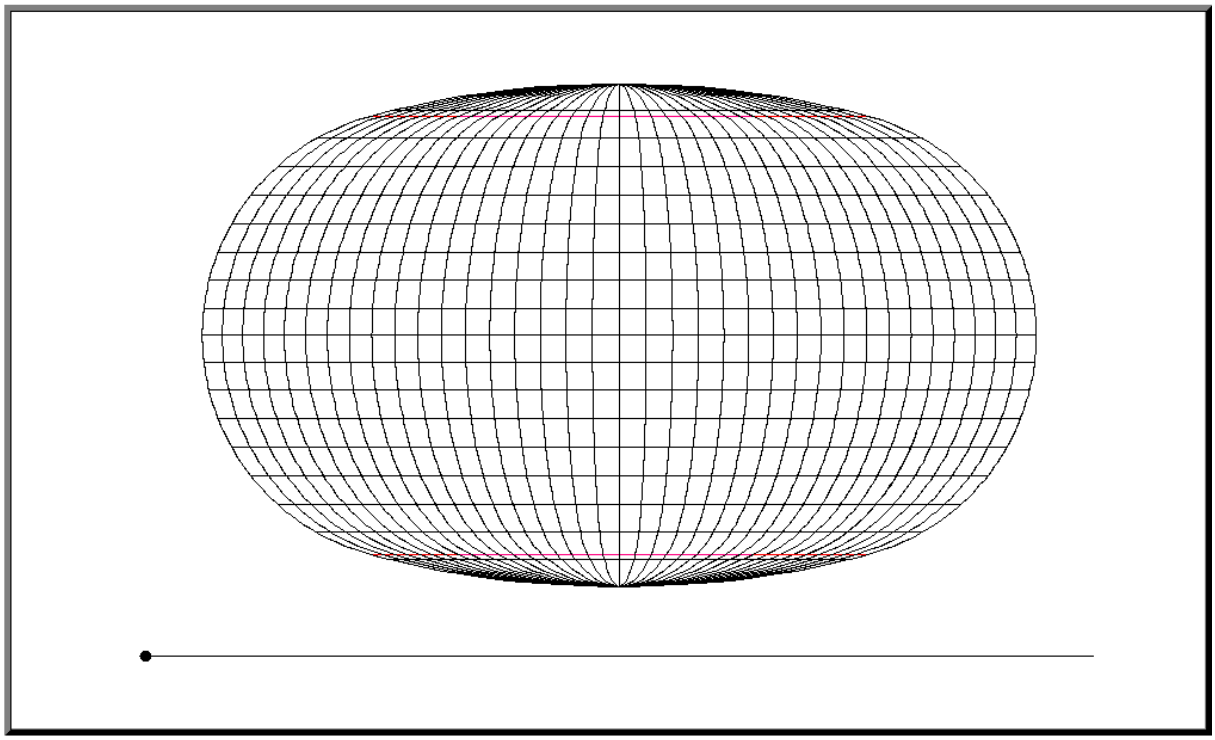
Megjelenítettem még az aktuális vetület normál-parallelkörét az alap-, és a képfelületen is.



20. ábra

2.2.8. Osztott Baranyi vetület

Az ebben a fejezetben bemutatásra kerülő program egy tanszéki vonatkozású vetületet dolgoz fel. A vetületet dr. Márton Mátyás alakította ki, alapul használva a Baranyi IV-es általános torzulású vetületet, matematikai leírását pedig dolgozatom témavezetője adta meg. A vetület lényege, hogy a világ óceánjait előnyösen ábrázoljuk a szárazföld rovására. Ez úgy valósult meg, hogy az eredeti összetett képzetes hengervetület két egymásba kapcsolódó, de külön vetületre osztódott. A két rész közötti kapcsolatot egy 40°-os zóna biztosítja, amelynek északi-félgömbi részét a jobb, a déli részét pedig a bal oldal tartalmazza. Ezek meridiánjai egymásba törésmentesen kapcsolódnak az egyenlítőnél. Mindkét résznek más délkör adja a középmeridiánját. Ezeken felül még egy a Jeges-tengert ábrázoló kis melléktérkép is megjelenik a térképműben, amely a 2.2.5.-ben szemléltetett Postel-féle síkvetületben készült. A programot a témavezető honlapján 2010 októbere környékén elérhető vetületi egyenletek segítségével szerkesztettem, de ezzel a megjelenítési technikával volt szerencsém segíteni a témavezetőt a vetület végleges leírásának megalkotásában. Az egyenletek két alakja egyenlő egymással. Szakdolgozatomban ez a második animáció. A vetület készítésének folyamatát mutatja be, viszont kommentár nélkül nem valószínű, hogy ez az illusztráció képes megértetni a gondolatmenetet. A témavezető ötlete volt a program elkészítése, én nem is ismertem a vetületet azelőtt. Az animáció állomásokból áll, amelyeket a témavezető határozott meg. Az ezek közötti átmenet szerkesztéséhez szabad kezet kaptam, így ezeket próbáltam látványosra megalkotni. A kezdő momentum az alapvetület, a Baranyi IV bemutatása fokhálózati vonalakkal. A második állomás a kettéosztódott állapotot mutatja be. Ekkor látható, hogy a két rész még nem kapcsolódik egymáshoz megfelelően, amit a piros villogó meridiánvégek hivatnak jelezni. Egy transzformációval értem el a kapcsolódás folyamatos javulását és beteljesülését, amit a villogás frekvenciájának növekedésével és színváltással észlelünk. Következő állomás a Tazmániát tartalmazó zónarészlet megjelenése, ami az átmeneti rész kiterjesztése, majd ezután a síkvetületű melléktérkép. A legvégső állomás pedig az egész vetületet körülfoglaló kontúrvonal, a kontinensek körvonalai, és a cím megjelenése.



21. ábra

- **Szerkesztés**

Talán ez vette el a legtöbb munkaórát, mivel nagyon sok kis egységből állt a szerkesztés. Egységnek tekintem a vetület azon tartományát, ahol ugyanazokat az egyenleteket alkalmazom. Mivel a program elkezdésénél még nyitva hagytam azt a lehetőséget, hogy majd az applet-et nemcsak animációként, hanem interaktívan is tudjuk kezelni, ezért nem az optimális polyline-okból, hanem line-okból épül fel, növelve a számítások mennyiségét. Azért lett volna probléma a polyline-okkal, mert nagyításnál eltűntek volna. Olyan opciót se zártam ki, hogy a vetületi torzulásokat a ki nem használt **Z** koordináták segítségével szemléltessem, de végül ezeket elvetettem.

Az első megoldandó akadály az állomások összekapcsolása volt. Azt választottam megoldásnak, hogy az egyik állomásból folyamatos átalakulással megjelenjen a másik. Én ezt egy nem tudom, hogy mennyire sajátos módon valósítottam meg: Az applet-ben látható vonalak pontokból állnak. A transzformáció elején és a végén ismert a helyük. Keresek egy görbét, amely mentén elmozgatom a pontot, és amelynek ismerem két határozott pontjában a paramétere értékét, mivel ezzel a paraméterrel tudom irányítani majd a transzformáció

menetét. Kézenfekvő és egyszerű megoldás az egyenes, melynek egyenletrendszere egy pont első és második állomásbeli helye segítségével a következő:

$$\vec{\mathbf{E}}(\mathbf{g}) := \vec{\mathbf{P}}_1 + \left(\vec{\mathbf{P}}_2 - \vec{\mathbf{P}}_1\right)\cdot \mathbf{g} \quad \textbf{ahol} \quad \mathbf{0} \le \mathbf{g} \le \mathbf{1}$$

Tehát **g** paraméter változtatásával egy pontot el tudok mozgatni a helyéről egy egyenes mentén, és így **g**-t 1-el egyenlővé téve egy előre meghatározott helyre tehetem. Egy vonal összes pontjára alkalmazva ugyanazt a függvényt pedig az egész görbét így eltranszformálhatom a kívánt alakzatba egy másik helyre. $\vec{\mathbf{P}}$ helyére írva a megfelelő vetületi egyenleteket létrehoztam az animációban látott vonalakat (például az átmeneti zóna déli része). Vannak olyan görbék, amelyek többször is áttranszformálódnak az animáció során. Ezeket egymás után két különböző egyenesen tolom el, aminek leírásához $\vec{\mathbf{E}}(\mathbf{g})$ -t beágyazom egy másik egyenes egyenletébe:

$$\mathbf{F}(\mathbf{g},\mathbf{h}) := \vec{\mathbf{E}} + \left(\vec{\mathbf{P}}_3 - \vec{\mathbf{E}}\right)\cdot \mathbf{h}$$

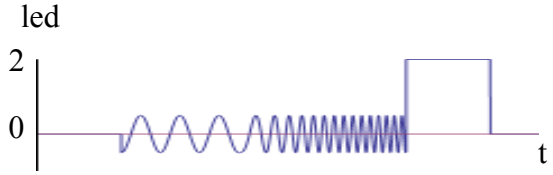
Ezt például az átmeneti zóna északi részénél alkalmaztam.

Az ezeket mozgató paramétereket (**g** és **h**) egy alkalmasan megválasztott időfüggvény adja. Ahhoz, hogy a transzformáció gördülékenynek tűnjön, ennek a függvénynek törésmentesnek kell lennie. További kritérium, hogy a transzformáció kezdetén a függvény értékének 0-nak, a végénél pedig 1-nek kell lennie. Így megfogalmaztam az alábbi függvényt:

$$\mathbf{g}(\mathbf{t}) := \begin{cases} \mathbf{0} & \textbf{ha} \quad \mathbf{t} \le \mathbf{t}_1 \ ; \\ \mathbf{1} & \textbf{ha} \quad \mathbf{t} \ge \mathbf{t}_2 \ ; \\ \dfrac{\mathbf{1} - \cos\left(\dfrac{(\mathbf{t}-\mathbf{t}_1)\cdot\pi}{\mathbf{t}_2-\mathbf{t}_1}\right)}{\mathbf{2}} & \textbf{ha} \quad \mathbf{t}_1 < \mathbf{t} < \mathbf{t}_2 \end{cases}$$

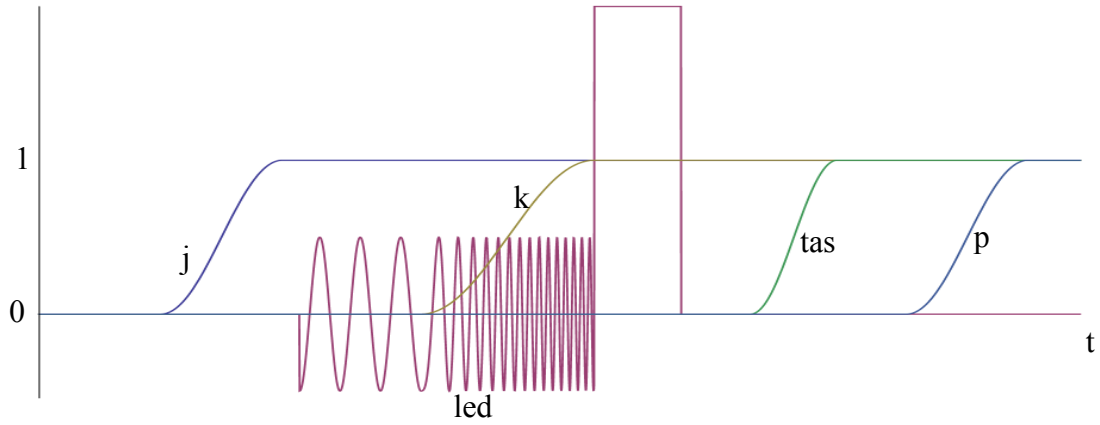
Természetesen minden transzformációhoz külön megszabtam egy paramétert, egy kezdő ($\mathbf{t_1}$) és egy befejező időpontot ($\mathbf{t_2}$).

A második és a harmadik állomás közötti meridiánvég-villogást úgy oldottam meg, hogy a különböző színű vonalakat egy időben változó értéktől függően megjelenítem. Ezt az értéket az alábbi függvénygrafikon szemlélteti:



$$\vec{\mathbf{L}}(\mathbf{t}) := \begin{cases} \textbf{piros} \quad \textbf{ha} \quad \mathbf{0} \le \textbf{led} \le \mathbf{1} \\ \textbf{zöld} \quad \textbf{ha} \quad \textbf{led} = \mathbf{2} \end{cases}$$

A legutóbbi villogásért felelős és a transzformációkat mozgató paraméterek egy összegző függvénygrafikonban jól áttekinthetőek:



Az ötödik állomást, azaz a melléktérképet megjeleníteni bonyodalmasabb volt mint hittem volna. A vetületi leírásban az szerepel, hogy a kontúrkör középpontja és a vetületi kezdőpont nem esik egybe. Ez nem CorelDraw, ahol csak úgy levágjuk a felesleges részt, itt meg kell mindent fogalmazni a matematika nyelvén. A meridiánokat a következőképpen csináltam: Megoldottam a következő egyenletet **β**-ra, amely azt takarja, hogy megkeresem az origóba helyezett vetület meridiánjainak, és az attól az x tengelyen negatív irányba eltolt körnek a metszéspontjait:

$$\beta\cdot\sin(\lambda)=\sqrt{r^2-(\beta\cdot\cos(\lambda)+dist)^2} \qquad \text{ahol} \quad dist:=\text{a két középpont távolsága}$$

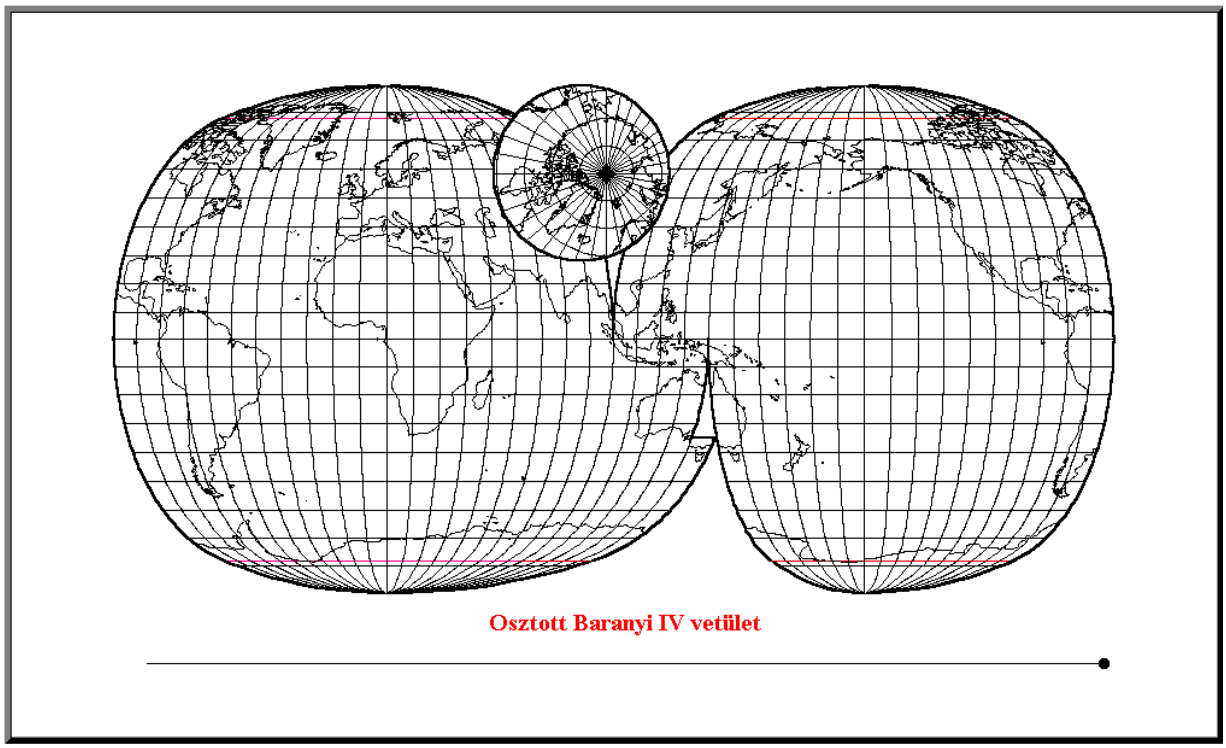
Ezután a kapott **β**-t felhasználom a vetület egyenleteiben (2.2.5.), megszorzom **p**-vel, amely a transzformációért felelős paraméter, és eltolom a térkép síkján a megfelelő helyre. Ezeket a pontokat páronként beteszem egy-egy **Line[..]** parancsba, és már el is készült. A parallelköröket kicsit más módszerrel csináltam: Megcsináltam **β=10°**-tól **60°**-ig a parallelköröket apró **Line[..]**-okból felépítve, majd ezekből a fentebbi egyenlethez hasonló egyenlőtlenséggel leválogattam a körön kívülre esőket. Emiatt az illeszkedés a határvonalon nem pontos, de ez nem látszik a méretarány miatt.

Még egy új elem található az applet-ben, mégpedig a kontinensek kontúrvonalainak megjelenése. Azt találtam ki, hogy a vonalak ne egyszerre, hanem folyamatosan, szegmensekként jelenjenek meg. Ezt úgy csináltam, hogy a vonalakat a processzor örömére megint csak **Line[..]**-okból építettem fel, sőt minden vonalhoz egy külön feltételt rendeltem. Ezek a feltételek azt mondják, hogy ha az időparaméter (**t**) egy megszabott értéknél nagyobb,

akkor jelenjen meg a vonal. A vonalakhoz ezeket az értékeket a parancsok generálásakor rendeltem hozzá, amelyek egy **r1** és **r2** időintervallum között egyenletes eloszlás szerint helyezkednek el. A kontinensek megjelenítése több időt vett el, mint az eddigiek, mivel itt elég hosszú és bonyolult parancsokat kellett elsajátítani és használni. Ezek a parancsok az alap adatsor átalakítására, abból való bányászatra és új adatsor kreálására valóak.

Fontos dolog, hogy a vetület alkalmazásánál az egyenletekben szereplő **Δλ**-t mindig a 2.2.3.-ban megfogalmazott módon definiáljuk, tehát szorítsuk -180° és 180° közé, hogy jó helyre kerüljön a térképi tartalom.

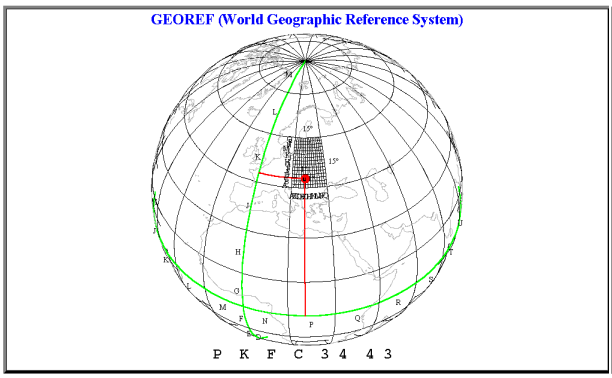
Legvégül pedig megjelenik az animáció címe, amely egyben egy link egy raszteres képhez, amely az eredeti Osztott Baranyi vetületet használó térképműnek egy kisebb felbontású változata.



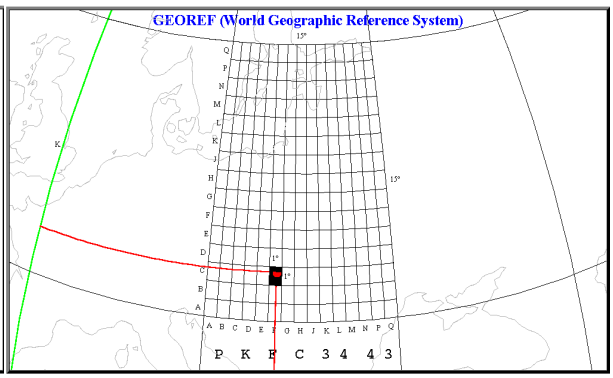
22. ábra

2.2.9. GEOREF

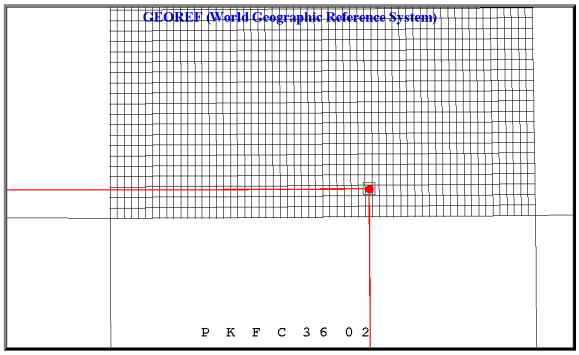
A legutolsó applet tárgya egy globális területazonosító-rendszer, amely a Föld valódi fokhálózatára épül. Vannak olyan azonosító-rendszerek, amelyek a Föld vetített alakjára építenek (az UTM vetületű MGRS), ezeket általában helyi katonai használatra tervezik. Van olyan is, amely alapja a címadó rendszerhez hasonlóan „lat/lon", viszont kódjuk nagy mértékben optimalizálva van (Natural Area Code, amelyet pl. nemzetközi postai azonosításra használnak). A World Geographic Reference System (GEOREF)-et USAF számára tervezték globális alkalmazásra. Alfanumerikus azonosító karakterei az aktuális terület bal alsó sarokpontjára vonatkoznak. Az első karakter a 15°-os beosztású fokhálózaton definiál egy gömbkétszöget betűvel. A sorozat **-180°**-tól indul „**A**"-val, és „**Z**"-vel végződik **+165°**-nál úgy, hogy „**I**"-t és „**O**"-t kihagyja. Ugyanígy a második karakter „**A**"-tól „**M**"-ig **15°**-os foktrapézokra osztja a kétszöget D-É irányítottsággal. Az alsóbb szintekre is igaz, hogy a jelzés a bal alsó sarokból indul, és először hosszúság, majd szélesség szerint jelölődik. A második szint, azaz a harmadik és negyedik karakter a **15°**-os foktrapézt bontja **1°**-os rácsra betűkkel „**A**"-tól „**Q**"-ig (24. ábra). A harmadik szintje a jelölésnek már számokat használ, és a fokperceket jeleníti meg (25. ábra). Itt igénytől függően használhatunk perc (**PK FC *34 43***), tizedperc (**PK FC *348 432***) vagy századperc (**PK FC *3481 4326***) pontosságot is. Az elkészített program fokperc pontosságot használ. Van még lehetőség a magasság jelölésére is, de az nem jelenik meg az applet-ben. Az applet megjelenése hasonlít 2.2.3.-re, de itt csak egy pontot mozgathatunk a gömbön, amelynek viszont megjelenik a koordinátája. A GEOREF első szintjének (tehát a **15°**-os foktrapézoknak) koordinátavonalai közül az összes látszik. Az érthetőség kedvéért az egyenlítőre és a kezdőmeridiánra ráírtam az adott zóna és öv jelzését. A második szint koordinátavonalai, azaz az **1°**-os rács, az átláthatóság kedvéért csak a változtatható felületi pontot tartalmazó foktrapézon belül látható. A foktrapéz szélén kerültek megírásra a megfelelő koordináták és a foktrapéz mérete, de ezek már nem statikusak, hanem a ponttal együtt elmozdulnak, ha szükséges. Ugyanígy a harmadik szintből is csak az aktuális vonalak látszanak, viszont itt csak az **1°**-os foktrapéz méretéhez van megírás. A pontból két piros leolvasóvonal vezet az előbbiekben felvázolt megírásokhoz, mint egyfajta leolvasóléchez. Így láthatjuk, hogy honnan származnak a kijelzett koordináták, és nagy mértékben hozzájárul a megértéshez.



23. ábra



24. ábra



25. ábra

- **Szerkesztés**

A kontinensek körvonalait 2.2.3.-ból átmásoltam, a 15°-os fokhálózatból pedig kiemeltem az egyenlítőt és a kezdőmeridiánt, majd zöldre színeztem. Az ezeken lévő betűk **Text[..]** kiíratő parancsait **Table[..]** függvénnyel kreáltam, viszont magukat a betűket ezekbe kézzel írtam be. A felületi pontot a szokásos módon alkottam meg, a belőle kiinduló leolvasó vonalak egyenletei pedig a következőek:

$$\vec{L_1} := \begin{bmatrix} \cos(\varphi)\cdot\cos(g\cdot\lambda) \\ \cos(\varphi)\cdot\sin(g\cdot\lambda) \\ \sin(\varphi) \end{bmatrix}; \quad \vec{L_2} := \begin{bmatrix} \cos(g\cdot\varphi)\cdot\cos(\lambda) \\ \cos(g\cdot\varphi)\cdot\sin(\lambda) \\ \sin(g\cdot\varphi) \end{bmatrix} \quad \text{ahol} \quad \begin{matrix} \varphi := \arcsin(z); \\ \lambda := \arctan2(x,y); \\ 0 \le g \le 1 \end{matrix}$$

A továbbiakhoz szükség volt definiálni φ és λ kerekített értékeit:

A **15°**-ra kerekített értékek:
$$f2 := \text{floor}\left(\frac{\varphi}{15\cdot\text{deg}}\right)\cdot 15\cdot\text{deg}; \qquad l2 := \text{floor}\left(\frac{\lambda}{15\cdot\text{deg}}\right)\cdot 15\cdot\text{deg};$$

A **1°**-ra kerekített értékek:
$$f3 := \text{floor}\left(\frac{\varphi}{\text{deg}}\right)\cdot\text{deg}; \qquad l3 := \text{floor}\left(\frac{\lambda}{\text{deg}}\right)\cdot\text{deg} \qquad \text{ahol} \quad \text{deg} = \frac{\pi}{180}$$

Ezek segítségével az alsóbb szintek koordinátavonalait el tudjuk tolni.

Nem nagyon írtam a dolgozatba *Mathematica* parancsokat, amelyekkel a görbéket reprezentáló pont-sorozatokat generálom, de itt szükségesnek tartom bemutatását. A második szint, tehát egy **15°**-os foktrapézon belüli **1°**-os rács generálását az alábbi paranccsal végeztem:

```
Table[
    Line[
        Table[
            {Cos[ f2 + u*Degree]*Cos[ l2 + v*Degree],
            Cos[ f2 + u*Degree]*Sin[ l2 + v*Degree],
            Sin[ f2 + u*Degree]},
        {u, 0, 15}]
    ],
{v, 1, 14}]
```

Itt egy vonal argumentumába 16-szor belekerül ez a három koordináta úgy, hogy **u** helyére 0-tól 15-ig ír egész számokat, és ez megismétlődik 14-szer mindig más **v** értékkel. Tehát született 14 polyline. Ez még csak az É-D-i vonalakat adja, ezért ezután kicserélem az utolsó **u**-t és **v**-t, majd megint elvégzem a műveletet. A harmadik szintet hasonló módon csinálom, de mivel ezek a koordinátavonalak már fokpercek szerint következnek, ezért alkalmazni kellett egy átalakítófüggvényt, amely a „ _°_'_” ”-ből csinál „ _° ”-ot:

```
Table[
    Line[
        Table[
            {Cos[( f3/Degree + FromDMS[{0, u, 0}])*Degree]*
                Cos[( l3/Degree + FromDMS[{0, v, 0}])*Degree],
            Cos[( f3/Degree + FromDMS[{0, u, 0}])*Degree]*
                Sin[( l3/Degree + FromDMS[{0, v, 0}])*Degree],
            Sin[( f3/Degree + FromDMS[{0, u, 0}])*Degree]},
        {u, {0, 60}}]
    ],
{l, v, 59}]
```

Megjegyzem, hogy itt a belső **Table[..]** függvény kiértékelendő értékei között csak 0 és 60 szerepel. Ezáltal az nem megy végig az összes értéken, így a polyline helyett csak line lesz az

eredmény. Azért engedtem meg magamnak ezt az egyszerűsítést, mert ilyen kis távolságoknál a gömb felülete már közel sík, és nem észrevehető a különbség.

A koordináták kiíratásához szükség van még a pont helyzetének szögperc értékére. Ennek meghatározása a kiíratás később leírt nehézségéből kifolyólag számjegyenként történt:

$$f4 := \mathrm{floor}\left(\frac{\varphi - f3}{\mathrm{deg}} \cdot 6\right); \qquad f5 := \mathrm{floor}\left(\frac{\varphi - f3}{\mathrm{deg}} \cdot 60\right) - f4 \cdot 10$$

$$l4 := \mathrm{floor}\left(\frac{\lambda - l3}{\mathrm{deg}} \cdot 6\right); \qquad l5 := \mathrm{floor}\left(\frac{\lambda - l3}{\mathrm{deg}} \cdot 60\right) - l4 \cdot 10$$

Tehát a pontnak a kiírt koordinátái:

X(l2)X(f2) X(l3)X(f3) l4f4 l5f5

(pl.: EK PH 26 40)

Nem gondoltam volna, de az applet alsó részén látható aktuális koordináta szerkesztése több időt vett igénybe, mint az eddigiek. Így leírva nem nehéz az elv, amely alapján megcsináltam, de időbe telt, míg rájöttem a dolog nyitjára, mivel eddig nem szembesültem ilyen problémákkal. Három egymásból adódó probléma volt: 1. nem tudunk *string* típusú változókat használni, amelyeket betehetnénk egyszerűen a címkezelésért felelős **PlotLabel[..]** vagy a **Text[..]** parancsba, ezáltal ezeknek használata eléggé korlátozott. 2. Meg kell oldani, hogy a felirat mindig a képernyő egy fix helyén legyen. 3. A karakterek relatív távolsága se változzon. Megoldások: 1. Minden betűnek és számnak külön **Text[..]** parancsot fogalmaztam meg, és ezeket egy helyiértéken megjelenő összes lehetséges karakter számától függő mélységű **HA()** függvénybe tettem. 2. A betettem az origóba az összes szöveget, majd a képernyő síkjában eltoltam a kellő helyre (szerencsére van ilyen opció a **Text[..]** parancson belül). 3. A második probléma megoldásában benne van a hiba, mert az eltolást a betűméret alapján teszi a fordítócsomag, ezért ezt a betűtípus egyenközűsítésével (font: Monospaced) oldottam meg.

3. Összefoglalás

Úgy érzem, hogy elértem a célomat, mivel sikerült létrehozni olyan eszközöket, amelyek a földi és térképi koordináta-rendszerek témakörében egy-egy adott témát megfelelően bemutatnak és megfelelnek az általam felállított követelményeknek. Persze ez egyáltalán nem jelenti azt, hogy megcsináltam volna az összes lehetséges vizualizációt, hiszen milliónyi dolog van még amelyet meg lehet jeleníteni ilyen formában. Sok ötletünk maradt megvalósítatlanul a témavezetővel (pl.: MGRS animáció, EOV, kúpvetületek és a síkvetületek kapcsolata, sík- és hengervetületek szögtartóságának geometriai beláttatása, torzulási viszonyok, stb...), de ezek sajnos idő híján elmaradtak. Foglalkoztam még az egyes vetületeket bemutató programoknál a változó valódi fokhálózat megjelenítésével is, de ez technikai okok miatt nem került bele a dolgozatba. De ez nem is baj, hiszen erre létezik egy erre a célra tökéletesen megfelelő program [Bortolossi, 2007]. Remélem, hogy a későbbi szakdolgozók vagy akárki más, szakdolgozatomból merítve megpróbálkozik hasonló dolgokkal és létrehoz ezeknél színvonalasabb programokat. Nekik ajánlom, hogy próbálkozzanak majd az újabb, web-es standard-eknek megfelelő formátumokkal (pl.: .x3d). Ezúttal is szeretném ösztönözni a diákokat, a tanárokat és akármilyen felhasználót, hogy használják a programokat, hiszen ezért készültek. Ezen kívül javaslom az applet-ek bekerülését segédanyagként a vonatkozó téma elektronikus jegyzetéhez, esetleg órai bemutatásukat, persze ha az illetékesek megfelelőnek tartják azokat.

4. Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, Dr. Györffy Jánosnak, hogy észrevételeivel, tanácsaival és ötleteivel segítette szakdolgozatom elkészítését, és hogy az aktuálisan felmerülő problémákat mindig gördülékenyen meg tudtuk beszélni. A közvetett segítsége pedig abban nyilvánult meg, hogy a szakdolgozat végül is az általa átadott tudásnak egy apró részlete modern eszközökbe öntve.

5. Irodalomjegyzék

Györffy János: Elektronikus jegyzet
http://mercator.elte.hu/~gyorffy/jegyzete/jegyzete.html

Nykamp D. Q., 2004-2009: Multivariable calculus and vector analysis
http://www.math.umn.edu/~nykamp/m2374/readings/divcurl/

Kraus, M. : LiveGraphics3D
http://www.vis.uni-stuttgart.de/~kraus/LiveGraphics3D/

Rogness J. : Constructing Mathlets Quickly using LiveGraphics3D
http://www.math.umn.edu/~rogness/lg3d/page_Introduction.html

Dr.-Ing. E.h. Dr. **Biró Péter**, 2004: „Felsőgeodézia (BSc)" elektronikus jegyzet, Budapest,162.2.

Kratochvilla Krisztina, 2003/10: Az EOV-alapfelületek térbeli helyzetének vizsgálata, Geodézia és Kartográfia, Budapest

Lavelle S. M., 2006: Planning algorhythms
http://planning.cs.uiuc.edu/node102.html

http://www.physicsforums.com/showthread.php?t=173847

http://en.wikipedia.org/wiki/Natural_Area_Code

http://en.wikipedia.org/wiki/Lambert_azimuthal_equal-area_projection

Bortolossi H. J., 2007: Map projections
http://www.uff.br/mapprojections/mp_en.html