SZEGEDI TUDOMÁNYEGYETEM

Természettudományi és Informatikai Kar
Bolyai Intézet Geometria Tanszék

SZAKDOLGOZAT

Gömbi geometria a számítógépen

Csépe Zoltán
Informatika - matematika

Témavezető: Dr. Nagy Gábor Péter

2011

1. Tartalomjegyzék

2. Tartalmi összefoglaló

Dolgozatomban a gömbi geometria tanításához segítséget nyújtó program készítését és használatát mutatom be. Fontos, hogy a gyerekek az euklideszi geometrián kívül nemeuklideszi geometriával is megismerkedjenek és ezzel egy más szemlélet módot is kialakítsunk bennük. Valamint a szokványos a tananyagban szereplő ismereteken kívül, egy kis többlet tudással ruházzuk fel őket. A program elkészítése során a NetBeans fejlesztői környezetben JOGL és Java Swing segítségével végeztem a fejlesztést valamint a tesztelést. A program kezelhetősége szempontjából fontos volt a look&feel módszer alkalmazása ami arra utal , hogy az egyes elemek kinézete egyértelművé teszi a funkciójukat. A dolgozatban elkészítettem egy óravázlatát, amelyen bemutatom a program használatát párhuzamosan a Lénárt gömbbel.

Kulcsszavak: gömbi geometria, OpenGL, JOGL, Java Swing, Lénárt gömb

3. Bevezetés

A mai középiskolai oktatásban egyre inkább háttérbe szorul a geometriaoktatás, pedig saját tapasztalatom alapján a gyerekek nagyon szeretik. Ezáltal pedig nincs is lehetőség nem euklideszi geometria bemutatására. Pedig sok érdekes dolgot tapasztalhatnának meg a gyerekek. De miért is érdemes a nem euklideszi geometriák közül épp a gömbi geometriával megismerkedniük. Mivel a földön élünk, amely egy gömb így rögtön adódik, hogy ezzel foglalkozzunk. Pedig itt is nagy lehetősségek vannak a kísérletezésre, mely szerintem a geometria tanításánál nagyon fontos. A gyerekek sokkal könnyebben értik meg és látják át az összefüggéseket, mert például látja, hogy az egyik pontot így változtatom akkor az egész rendszer hogyan alakul. Erre pedig a számítógép nagyon jó lehetőségeket biztosít. Az euklideszi geometria bemutatására rengeteg program született már, az egyik legismertebb szerintem a GeoGebra. A gömbi geometria kísérletezés útján való megismerésére is készült már „analóg” eszköz. Ez nem más, mint a Lénárt féle rajzgömb-készlet. Mely áll egy gömbből, melyre rajzolhatunk. Egy gömbi vonalzóból, mellyel két pont távolságát tudjuk meghatározni. Egy gömbi szögmérőből mellyel két gömbi egyenes által bezárt szöget adja meg. Továbbá egy gömbi körzőből, amely segítségével tetszőleges sugarú gömbi kört lehet rajzolni a gömb felületére. Ez a készlet ideális a gyerekek kísérletezéséhez, de hátránya, hogy nem tudják haza vinni az iskolából. Ezt próbáltam meg áthidalni, hogy elkészítem egy egyszerűsített változatát a rajzgömb-készletnek. Melyet a gyerekek akár otthon is tudnak használni. Ezáltal a saját ötleteiket is meg tudják valósítani. A programban lehetőség van pontok elhelyezésére, azokra gömbi egyenesek illesztésére. Pontok távolságának meghatározására. Valamint 3 pont által meghatározott szögek értékeinek meghatározására. Ezáltal önállóan figyelhetnek meg analógiákat az euklideszi geometriával. Dolgozatomban szeretném bemutatni a fejlesztés fázisait, tapasztalatokat és tovább lépési lehetőségeket. A rajzgömb készlet alap elemei közül a gömbi körzőt nem valósítottam csak meg. Fontosnak tartottam a könnyű, egyszerű kezelhetőséget. Az egyetemi tanulmányaim során több lehetőséget is láttam a középiskolásoknak a törzsanyagon túlmutató ismeretek bemutatására. De a számítógépes grafika kurzuson megismert dolgok mutatták a leglátványosabb lehetőségeket.

4. A gömbi geometria

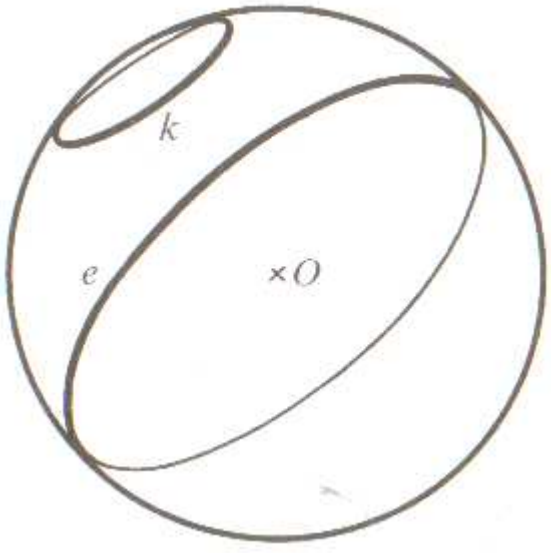
A fejezetben felhasznált irodalom:

- KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002,
- G. HORVÁTH Á. – SZIRMAI J.: Nemeuklideszi geometriák modelljei, Typotex kiadó 2004.

Mint az euklideszi síkon ugyanúgy végezhetünk vizsgálatokat a gömb felszínén. Melynek meglepő eredményei vannak a mindennapi életben is. Leggyakrabban a hajóskapitányok, repülőgép pilóták alkalmazzák. Meglepő, hogy Floridától a Fülöp-szigetek délre vannak, mégis ha repülővel megyünk Alaszkán keresztül rövidebb az út. Ennek az oka nem más mint, hogy Alaszka, Florida és a Fülöp-szigetek egy gömbi egyenesen vannak.

4.1. A gömbi geometria elemei

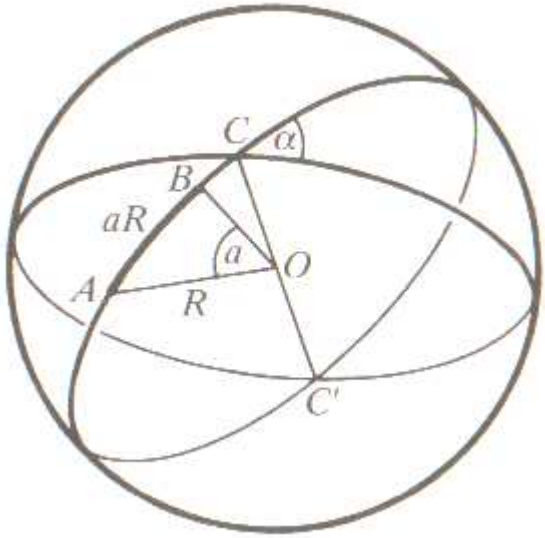
A gömbön is, mint a síkon a legegyszerűbb elem a *pont*. A síkon két pontra egy egyenes illeszkedik, a gömbön pedig két pontra egy *e főkör* (1. ábra). A főköröket olyan síkok metszik ki a gömbfelületből, amelyek illeszkednek a gömb középpontjára.



1. ábra Főkör és gömbi kiskör

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

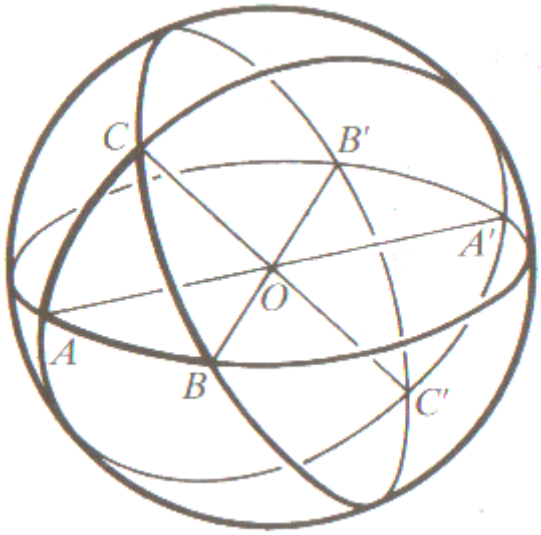
A körök szerepét a *k gömbi kiskörök* veszik át (1. ábra). Melyeket olyan síkok metszenek ki a gömbfelszínből, amelyek nem illeszkednek a gömb középpontjára. A gömbi szakaszok a félkörnél nem nagyobb főkörívek. Két pont *távolsága* a hozzájuk tartozó sugarak hajlásszöge (2. ábra). Melynek ha az ívmértékben vett hajlásszögét megszorozzuk a gömb sugarával akkor a két ponthoz tartozó főkör nem hosszabb ívének hosszát kapjuk. Tehát a két pont távolsága az őket összekötő főkörív hossza. Két főkör *hajlásszöge* pedig nem más, mint a hozzájuk tartozó síkok hajlásszöge (2. ábra).



2. ábra Két pont távolsága és két főkör hajlásszöge

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

A síkkal szemben a gömbön nem a háromszög a legegyszerűbb poligon. Hanem a *gömbkétszög* melynek két csúcsa van és oldalai két fél főkör. Tehát a gömbkétszöget két sík metszi ki a gömbfelületből. Két főkör a gömbfelületet négy gömbkétszögre bontja. De a gömbön is megtalálható a háromszög, melyet *gömbháromszögnek* nevezünk (3. ábra). Melyet a három pont és a rájuk illeszkedő főkörívek határoznak meg. Három főkör egy gömbfelszínt nyolc gömbháromszögre bontja. Mivel minden gömbháromszöghöz tartozik egy konvex triéder, mely meghatározza a csúcsait. Így kijelenthető, hogy a gömbháromszög szögei és oldali π-nél kisebbek. A jelölések a síkháromszögnél megszokott módon történek. Az oldalakat kisbetűvel a szögeket görög betűkkel, és az a oldallal szemben az α szög van.



3. ábra Gömbháromszög

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

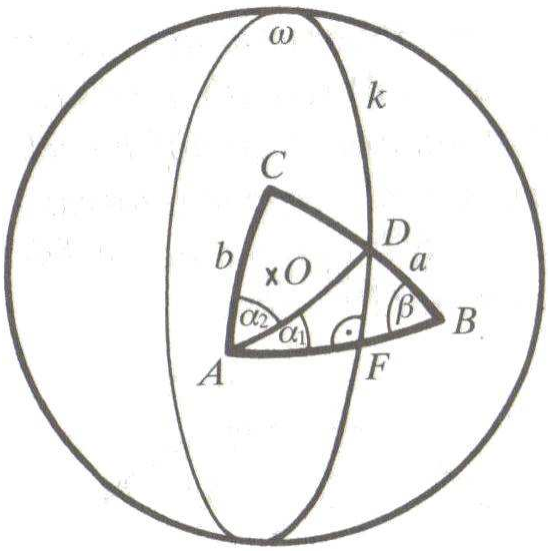
A tételek hasonlóan adódnak a gömbön, is mint a síkon.

Tétel: Egy AB főkörívet merőlegesen felező *k* főkör pontjai az A-tól és B-től egyenlő távolságra vannak. A *k* főkör által határolt félgömbfelületek belső pontjai pedig az AB főkörívnek attól a végpontjától vannak kisebb gömbi távolságban, amelyik ugyanahhoz a félgömbhöz tartozik, mint a vizsgált pont.

Tétel: Egy gömbháromszögben két oldal és az ezekkel szemközti szögek vagy páronként egyenlők, vagy nem egyenlők és akkor a hosszabb oldallal szemben a nagyobb szög van.

Bizonyítás: A 4. ábrán vegyük az α és β szögeket és a szemközti *a* és *b* oldalt. Ha az AB oldalt merőlegesen felező *k* főkörre illeszkedik a C csúcs akkor a gömbháromszög szimmetrikus, egyenlőszárú. Tehát $\alpha=\beta$ és $a=b$.

Ha a C pont az A-t tartalmazó *k* főkör által határolt félgömb felszínén van akkor $a>b$ az előző tétel alapján, így csak azt kell belátnunk $\alpha>\beta$. Az *a* oldalt a *k* főkör egy D pontban metszi, mivel B és C különböző félgömbön van. Az AD főkörív két részre bontja az α szöget, $\alpha_1+\alpha_2=\alpha$. A *k* főkör ω síkjára vonatkozó szimmetriából az is következik, hogy $\alpha_1=\beta$, tehát az α_1 szöget részként tartalmazó α szögnél a β szög valóban kisebb.

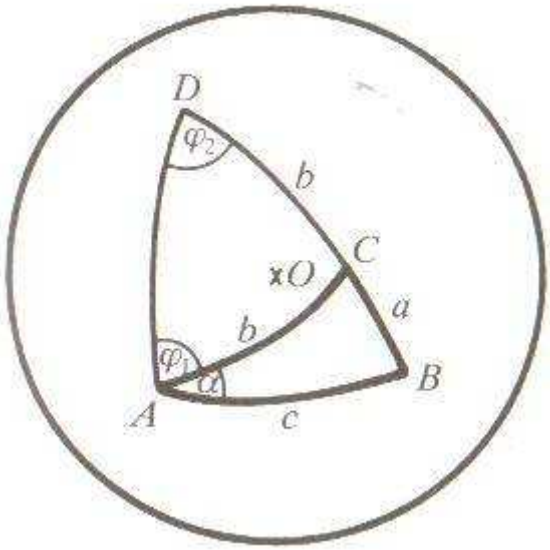


4. ábra A gömbháromszög szögeinek és oldalainak kapcsolata

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

Tétel: Egy gömbháromszög bármely két oldalának összege nagyobb a harmadiknál.

Bizonyítás: Ha a két oldal összege nagyobb vagy egyenlő, mint π akkor a tétel állítása nyilvánvaló. Mivel a gömbháromszög egyik oldala sem lehet nagyobb, mint π. Érdekesebb, azaz eset mikor két oldal összege nem nagyobb, mint π. Az 5. ábraán, ha BC oldalt meghosszabbítjuk az AC oldallal, akkor a BD oldal kisebb, mint π. Az ABD gömbháromszögnek a BD ív egy oldala, az AC ív pedig két részre bontja ezt a gömbháromszöget. Mivel DC=AC így az ACD gömbháromszögben a φ_1 és a φ_2 szögek egyenlők. Ezért az ABD gömbháromszögben az A csúcsnál lévő $\varphi_1+\alpha$ szög nagyobb, mint a D csúcsnál lévő φ_2. Valamint az előző tétel alapján a nagyobb oldal szemben nagyobb szög van, így $a+b > c$.



5. ábra A háromszög oldalainak összege

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

Tétel: Két gömbháromszög egybevágó, ha bennük páronként egyenlő:

- három oldal;
- három szög;
- két oldal és az általuk közre zárt szög;
- két szög és a csúcsaikat összekötő oldal;
- két különböző szinuszú oldal és a nagyobb szinuszúval szemközti szög;
- két különböző szinuszú szög és a nagyobb szinuszúval szemközti oldal;
- két egyenlő szinuszú de $\frac{\pi}{2}$-től különböző oldal és a valamelyikkel szemközti szög;
- két egyenlő szinuszú de $\frac{\pi}{2}$-től különböző szög és a valamelyikkel szemközti oldal;

A gömbön a legegyszerűbb alakzat a gömbkétszög melynek felszíne nagyon könnyen kiszámítható.

Tétel: Egy gömb gömbkétszögeinek a területe szögeikkel arányos.

Bizonyítás: Tekintsünk két olyan gömbkétszöget amelyeknek a megfelelő csúcsai azonosak. A C és a C' sarkpontokhoz tartozó egyenlítőnek, a gömbkétszögekkel vett metszéspontjai A_1 és A_2 valamint B_1 és B_2. F_α és F_β jelöli az α illetve a β szögű gömbkétszögek felszínét. Az α szöget osszuk fel n egyenlő részre, majd az $\frac{\alpha}{n}$ szöget mérjük fel a β szögre. Legyen $k \in \mathbf{N}$ az a szám, melyre

$$k \cdot \frac{\alpha}{n} \le \beta < (k+1) \cdot \frac{\alpha}{n}.$$

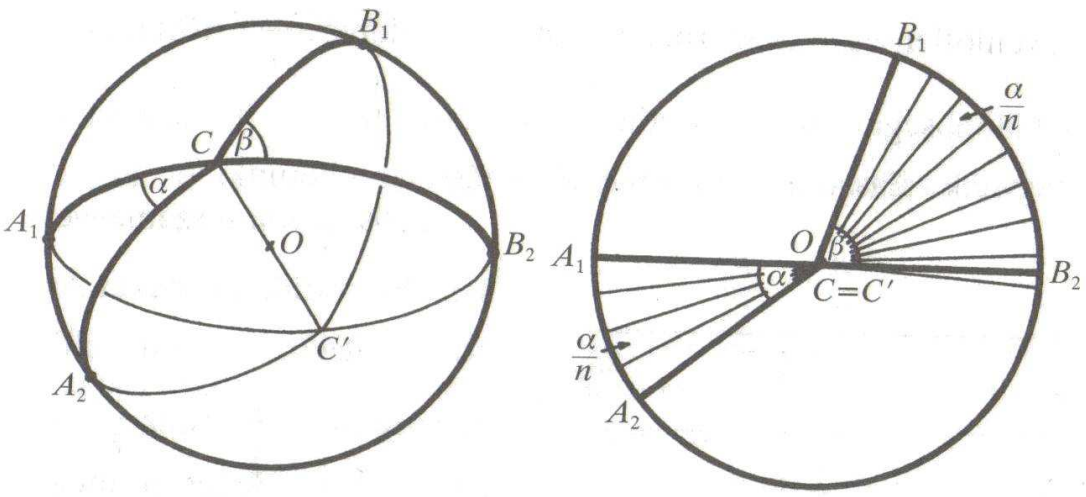
Ez a felosztás meghatározza a gömbkétszögek felszínének is egy felosztását. Az egyenlő szögekhez egyenlő felszínű gömbkétszögek tartoznak, tehát

$$k \cdot \frac{F_\alpha}{n} \le F_\beta < (k+1) \cdot \frac{F_\alpha}{n}.$$

Ezen egyenletekből α-val és F_α-val való osztás után:

$$\frac{k}{n} \le \frac{\beta}{\alpha} < \frac{k+1}{n} \text{ és } \frac{k}{n} \le \frac{F_\beta}{F_\alpha} < \frac{k+1}{n}.$$

Ekkor látszik hogy a $\frac{\beta}{\alpha}$ és $\frac{F_\beta}{F_\alpha}$ kisebb mint $\frac{1}{n}$ de nagyobb mint 0 és a 6. ábrán látható módon helyezkednek el.



6. ábra A gömkétszögek és felülnézetük

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

Be kell látnunk, hogy $\frac{\beta}{\alpha}$ és $\frac{F_\beta}{F_\alpha}$ egyenlők. Tegyük fel, hogy

$$\left| \frac{F_\beta}{F_\alpha} - \frac{\beta}{\alpha} \right| = \delta > 0,$$

ekkor ha *n*-et elég nagynak választjuk, akkor az $\frac{1}{n} < \delta$. Ekkor nem eshetne az $\frac{\beta}{\alpha}$ és $\frac{F_\beta}{F_\alpha}$ az $\frac{1}{n}$ hosszúságú szakaszra. Így ellentmondáshoz jutunk, így $\frac{\beta}{\alpha} = \frac{F_\beta}{F_\alpha}$.

Tétel: A *r* sugarú gömb α szögű gömbkétszögének a felszíne $2 \cdot \alpha \cdot r^2$.

Bizonyítás: Válasszuk az α szögű gömbkétszög mellé a π szögű gömbkétszöget. Mivel a gömb felszíne $4\pi r^2$ így a π szögű gömbkétszög felszíne $2\pi r^2$. Ekkor az előző tétel alapján $\frac{\alpha}{\pi} = \frac{F_\alpha}{2\pi r^2}$ amiből pedig adódik hogy $F_\alpha = 2\alpha r^2$.

Egy érdekes tétel a gömbháromszögre mely a síktól különbözik.

Tétel: A gömbháromszög szögeinek összege nagyobb, mint π és a felszíne $(\alpha + \beta + \gamma - \pi) \cdot r^2$.

Bizonyítás: Vegyük az ABC háromszöget a ábrán, melynek szögei α, β, γ. Az α szög meghatároz két gömbkétszöget, melyek tartalmazzák az ABC háromszöget és egy azzal egybevágó háromszöget. Valamint β és γ szöghöz is tartozik két-két gömbkétszög, melyek tartalmazzák a két gömbháromszöget. A gömböt lefedi a hat gömbkétszög, úgy hogy a háromszögeket háromszorosan. Így a gömbkétszögek felszínének képlete alapján:

$$2(2r^2\alpha + 2r^2\beta + 2r^2\gamma) = 4r^2\pi + 4T,$$

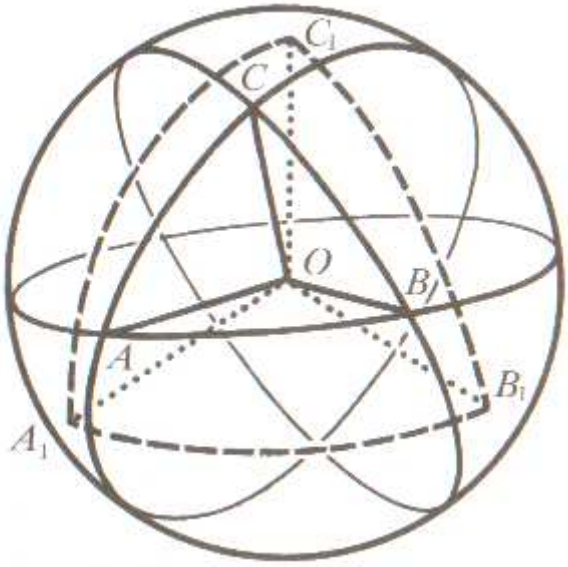
ahol a T az ABC gömbháromszög területe és átrendezéssel adódik, hogy

$$T = r^2(\alpha + \beta + \gamma - \pi).$$

Azt a szöget, amennyivel a gömbháromszög szögösszege meghaladja a π-t, gömbi feleslegnek nevezzük, ε-nal jelöljük. Így a gömbháromszög felszíne $\varepsilon \cdot r^2$.

Ha merőleges egyeneseket állítunk a gömbháromszög oldalainak a síkjára a gömb középpontján keresztül, akkor ezek a merőleges egyenesek két átellenes pontban metszik a gömbfelületet. Az A_1 B_1 ésC_1 jelöli az átellenes pontok közül azokat, amelyeket a hozzájuk tartozó oldal síkja nem választ el a gömbháromszögtől, amelyeket tehát a gömbháromszög harmadik csúcsával $\frac{\pi}{2}$-nél kisebb főkörív köt össze $\left(AA_1 < \frac{\pi}{2}; BB_1 < \frac{\pi}{2}; CC_1 < \frac{\pi}{2} \right)$. Az így kapott három pont $(A_1; B_1; C_1)$ ismét egy gömbháromszöget határoz meg. Az $A_1B_1C_1$ gömbháromszöget az ABC gömbháromszög polárgömbháromszögének nevezzük (7. ábra Polárgömbháromszög).

Ha egy k főkör ω síkjára merőleges átmérő a gömböt az A_1 és az A_2 átellenes pontokban metszi, akkor az A_1és az A_2 a főkör két pólusa, a k főkör pedig az $A_1; A_2$ pontok polárisa. Bármely gömbháromszög a saját polárgömbháromszögének a polárgömbháromszöge.



7. ábra Polárgömbháromszög

Forrás: KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

5. Alkalmazott technológiák

A fejezetben felhasznált irodalom:

- FEKETE Á. ZS.: Másodrendű felületek ábrázolása, Debreceni Egyetem szakdolgozat 2010,
- KUBA A.: Az OpenGL grafikai rendszer, Egyetemi jegyzet,
- NetBeans leírás
- Wikipedia – Java OpenGL

5.1. OpenGL

Az OpenGL a Silicon Graphics nevű cég által 1992-ben kifejlesztett szoftver interfész, mely a saját grafikus munkaállomásainak programozására kitalált IrisGL továbbfejlesztett változata. Az egyik legfontosabb szempont az OpenGL kifejlesztésénél a hordozhatóság volt, ezért az OpenGL, az IrisGL-el szemben más rendszerek felé is nyitott (OpenGL = Open Graphics Library). Ez a szoftver interfész pár száz eljárásból és függvényből áll, melyek lehetővé teszik 2 és 3 dimenziós grafikai objektumok létrehozását, és ezeken az objektumokon műveletek elvégzését. Az OpenGL tehát egy eljárás- és függvénygyűjtemény, melyek 2 és 3 dimenziós geometriai objektumok specifikációját tartalmazzák; ezen kívül olyan eszközöket is nyújt, melyekkel szabályozni lehet ezen objektumok leképezését a kép pufferbe, amelyben az OpenGL az eredményként létrejövő képet tárolja. Ennek megjelenítése már az operációs rendszer, vagy az ahhoz tartozó ablakozó rendszer feladata. Az OpenGL nem tartalmaz ablakozó rendszert, és nem támogatja az input eszközök kezelését sem, tehát ezeket a dolgokat az adott nyelven a programozónak kell megoldania. Jelen esetben erre szolgál a Java Swing. Az OpenGL-ben lehetőség van, a szintér definiálására háromdimenziós primitívekkel, a nézőpont meghatározására, megvilágítási modellek alkalmazására, árnyékok és textúrák alkalmazására, elsimításra, köd és további atmoszféra effektusok kezelésére. Az OpenGL primitíveknek nevezett grafikai alapelemeket rajzol. Ilyen primitívek a sokszögek, vonalak, pontok. A geometriai primitíveket vertexek definiálják. Egy vertex lehet egy pont, egy szakasz végpontja, vagy egy poligon csúcspontja, tehát minden grafikus primitív leírható vertexekkel. A vertexek nem csak a koordinátákat tárolják, hanem azok színeit és hozzátartozó információkat is, tehát a vertexek struktúrák. Az OpenGL a megjelenítéskor a Descartes féle koordináta rendszert használja, tehát a bázis olyan vektorokból áll, melyek

mindegyike merőleges a többire. A koordinátákat a megszokott x, y, z hármassal jelöljük. Mivel a számítógépes grafikában leggyakrabban a jobbsodrású rendszerek használatosak, ezért az OpenGL is ezt használja. Jobbsodrású koordináta-rendszer esetén a (0, 0, 0) pontban van az origó, az x, y tengely pozitív része az origótól jobbra, ill. fölfelé található, a z tengely pozitív része a képernyőből kifelé mutat.

5.2. JOGL

A JOGL egy java csomag mely biztosítja a Java Virtual Machine-nak a hozzáférést az OpenGL-hez. Ennek segítségével Javas objektumorientált szemléletet, lehet alkalmazni hardveres gyorsítással a kettő és háromdimenziós alakzatok megjelenítésénél. A hardveres gyorsításra azért van szükség, mert a felhasználó által érzékelt teljesítmény sokkal szebb lesz a nagy grafikus tartalmú Java alkalmazások esetén.

5.2.1. A JOGL használata

Az OpenGL paraméterezéséhez sok esetben mutatókat kell használni, viszont a Java nem támogatja ezeket, így tömbök segítségével tudjuk megvalósítani. Mivel a JOGL jól integrálható az alapvető Java komponensekbe, így már a meglévő Java osztályokból öröklés útján definiálja a rajzolható felületeket. Ezáltal nagyobb szabadságot biztosít, ellenben például a LWJGL[1]-el ahol előre meghatározott módon valósul meg az ablakkezelés. Így az LWJGL-ben nem használhatjuk a Java Swing elemeket, szemben a JOGL-al. A JOGL natív hívásokkal valósítja meg az OpenGL elérését így előre telepíteni kell. A megjelenítéshez kell egy a GLEventListener interfészt megvalósító rajzoló osztály. Amelyben a következő absztrakt metódusokat kell megvalósítani:

- init(GLAutoDrawable drawable) – Az inicializálást valósítja meg. Itt végezhetünk előzetes beállításokat. Általában itt állítjuk be a fényviszonyokat, a nem látható részek levágását, az árnyékolási módot, és a háttér színét. Valamint a változók egy részének az inicializálása is itt történik meg.
- reshape(GLAutoDrawable drawable, int x, int y, int width, int height) – Ha az ablak mérte, pozíciója változik meg, akkor kerül meghívásra. Itt állítjuk be a vetítés típusát, nézőpontot.

[1] LWJGL – Lightweight Java Game Library, egy grafikai osztály melyeket java-s játékok fejlesztésénél a kettő és három dimenziós elemek megjelenítéséhez használhatjuk.

- display(GLAutoDrawable drawable) – A megjeleníteni kívánt dolgokat tartalmazza. Ez a metódus folyamatosan meghívódik más komponenseken keresztül.
- displayChanged(GLAutoDrawable drawable, boolean modeChanged, boolean deviceChanged) – Ha megjelenítő eszköz változik meg, akkor kerül meghívásra. Ha a megjelenítési mód változik meg akkor a modeChanged paraméter igaz értéket kap, ilyen lehet, ha felbontás változik vagy a színmélység. Ha a megjelenítő eszköz változik, akkor a deviceChanged paraméter kap éréket. Ilyen, ha egyszerre jelenítünk meg a monitoron és egy projektoron is dolgokat, és az ablakot az egyikről a másikra húzzuk. Viszont ennek a metódusnak nem szükséges megadni implementációt, mert ezt a referencia OpenGL implementációk sem tartalmazzák.

Továbbá kell egy panel amelyen megjelenítjük a kívánt elemeket, ez jelen esetben egy GLJPanel de lehetne GLJCanvas is. Ehhez a panelhez kell hozzákapcsolni a kirajzolást megvalósító osztályt, az addGLEventListener metódussal. Mely a kéréseknek megfelelő metódust hívja meg. Szükség van még egy Animator osztályra mely vezérli a hozzá kapcsolódó megjelenítendő elemek display() metódusait. Az Animator egy háttérben futó szál.

```
private GLJPanel panel;
private Animator animator;
GLRenderer gl = new GLRenderer();
panel.addGLEventListener(gl);
animator = new Animator(panel);
```

5.3. Java Swing

A program készítése közben előtérbe került a felhasználóbarát, könnyen kezelhető felület létrehozása. Ezért a már sokszor kipróbált, jól bevált Java Swing rendszert alkalmaztam. Az ezzel megvalósított eszközök a Look&Feel minta alapján működnek. Ami azt jelenti, hogy megjelenésükből következik a használatuk. Ezáltal a felhasználók könnyedén az egerük segítségével vezérelhetik a programot, mely sokkal egyszerűbb, mint billentyű parancsok segítségével.

5.3.1. Alkalmazott Java Swing elemek

- GLJPanel – JPanel-ből öröklődéssel létrehozott tároló, melyben az OpenGL-s elemeket hardveres gyorsítással tudjuk megjeleníteni.
- JButton – Nyomógomb, a legalapvetőbb vezérlő elem. Általában valaminek például adatbevitel lezárására szolgál.
- JCheckBox – Jelölő négyzet, melynek két értéke lehet. Jól alkalmazható különböző funkciók ki és bekapcsolására.
- JComboBox – Legördülő menü segítségével könnyen tudunk különböző elemek közül választani. Alapállapotban csak az aktuális elem látszik, legördítés után viszont az összes lehetőség láthatóvá válik.
- JLabel – Szöveget tudunk vele elhelyezni az ablak egy tetszőleges pontján.
- JTextField – Szöveges beviteli mező, mely nagyon egyszerűen alkalmazható a felhasználóval való kommunikációra. Hátránya, hogy hiába számot írunk be az String-ként kerül a programba, így megfelelő konverziót kell végezni.

A Java Swing további elemeket is biztosít a felhasználóval való kommunikációra, de úgy láttam, hogy a fentebb felsorolt elemek elegendők a könnyed kommunikációra. Bár nem a Java Swing része, de szervesen kapcsolódik a felhasználóval való kommunikációhoz az egér mozgásának lekezelése. Fontos szempont volt, mert a megjelenített gömböt egér segítségével lehet könnyen mozgatni. Ennek megvalósításához az OpenGL-es elemeket tartalmazó panelhez rendeltem hozzá egy MouseEventListenert, egy MouseMotionListenert és egy MouseWheelListenert. Melyek közül a MouseEventListener segítségével tudtam az egérrel való kattintást és a folyamatos gomb lenyomást lekezelni. A MouseMotionListenerrel lehetett az egérrel való vonszolást lekezelni, melynek a gömb mozgatásánál van szerepe. Végül pedig a MouseWheelListenerrel lehette az egér görgőjének a mozgását lekezelni, melyre a közelítés és távolítás van felépítve.

```
panel.addMouseListener(this);
panel.addMouseMotionListener(this);
panel.addMouseWheelListener(this);
```

A vonszolásos gömbmozgatás az alábbi módon lett megvalósítva a MouseMotionListenerrel. Ahol az ellenorzes() függvény állapítja meg, hogy megtett-e már a gömb egy teljes fordulatot.

```
public void mouseDragged(MouseEvent e) {
    if(button1){
```

```
            gl.yRot += 360*(e.getX()-movePrevX)/e.getComponent().getWidth();
            movePrevX = e.getX();
            gl.xRot += 360*(e.getY()-movePrevY)/e.getComponent().getHeight();
            movePrevY = e.getY();
            ellenorzes();
            panel.repaint();
        }
    }
```

5.4. NetBeans

A NetBeans egy nyílt forráskódú fejlesztői környezet. Olyan környezet mely lehetővé teszi programok írását, fordítását, tesztelését, majd a kész programok telepítését. Java nyelven készült, de bármilyen más programozási nyelven is tudunk fejleszteni. Ingyenes termék, és a használatára nincsenek korlátozások.

A NetBeans sok modullal bővíthető, ezek közül a JOGL bővítő modult használtam. Az ilyen projekt készítése esetén a NetBeans automatikusan generálja a megjelenítéshez szükséges ablakot és a rajzoláshoz szükséges panelt. Lehetőség van a Swing elemeket drag & drop technológiával elhelyezni egy design ablakban, így a fejlesztői környezet automatikus generálja a hozzájuk tartozó kódot. Ezzel rengeteg idő spórolható meg a fejlesztés során.

5.5. Elvetett technológiák

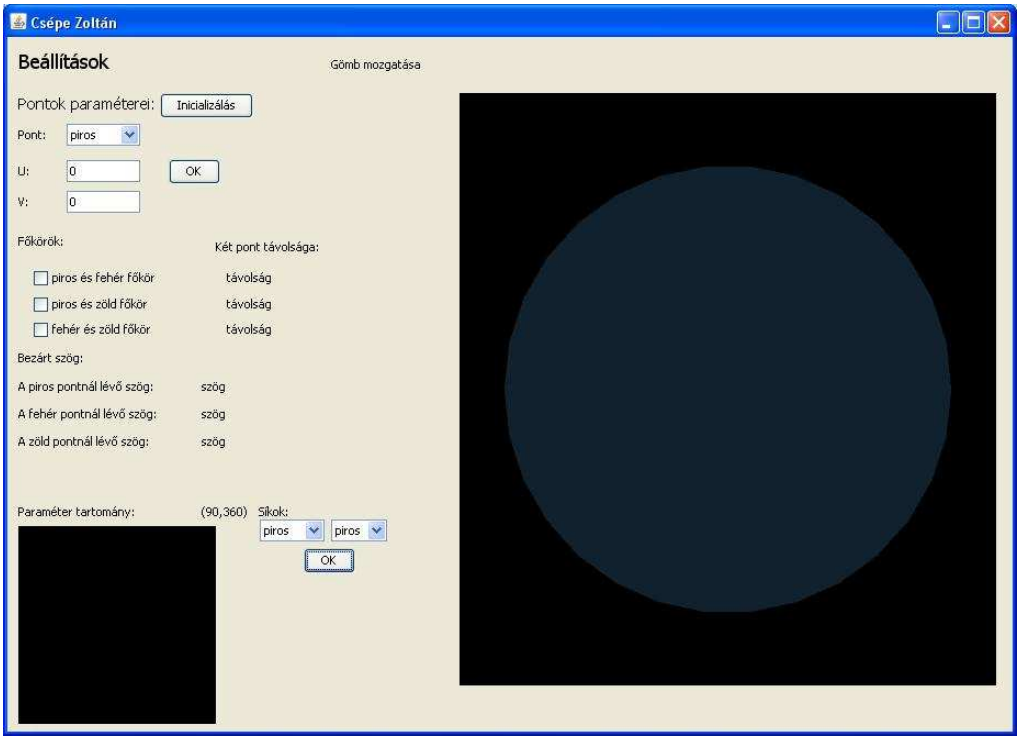
A fejlesztés kezdetén még nem a Java és a JOGL segítségével szerettem volna megoldani a szoftver fejlesztését. A fejlesztés elején még a GLUT segítségével C nyelven szerettem volna megoldani a problémákat. De idővel be kellett látnom túl költséges idő szempontjából, és nem is lesz a legszebben kinéző program. Valamint a kezelése is túl bonyolult és nehézkes lett volna. Nagy előnye lett volna a Java-s alapokkal szemben, hogy nincs szükség hozzá Java Virtual Machine-re. Viszont hátránya, hogy nehezebb hardveres gyorsítást létrehozni, és a már említett nehézkes kommunikáció a felhasználóval. Valamint a Java-s NetBeans fejlesztői környezet automatikus generálja az ablakot, amely részben rejtve is marad fejlesztés során. Ezzel szemben a GLUT-os környezetben az ablakot kézzel kell létrehozni, és nem lehet rá vezérlő elemeket létrehozni. Az egyetlen lehetőség a jobb klikkre előugró menü.

```
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
glutInitWindowSize(800, 600);
glutCreateWindow("Szakdoga");
glutReshapeFunc(ChangeSizeOrtho);
glutSpecialFunc(SpecialKeys);
glutKeyboardFunc(Keyboard);
glutDisplayFunc(RenderScene);
Menu();
SetupRC();
glutMainLoop();
```

6. A gömbi geometria számítógépes megjelenítése

6.1. A szoftver működése

Az indítás után az 8. ábraán látható képernyőkép fogad minket.



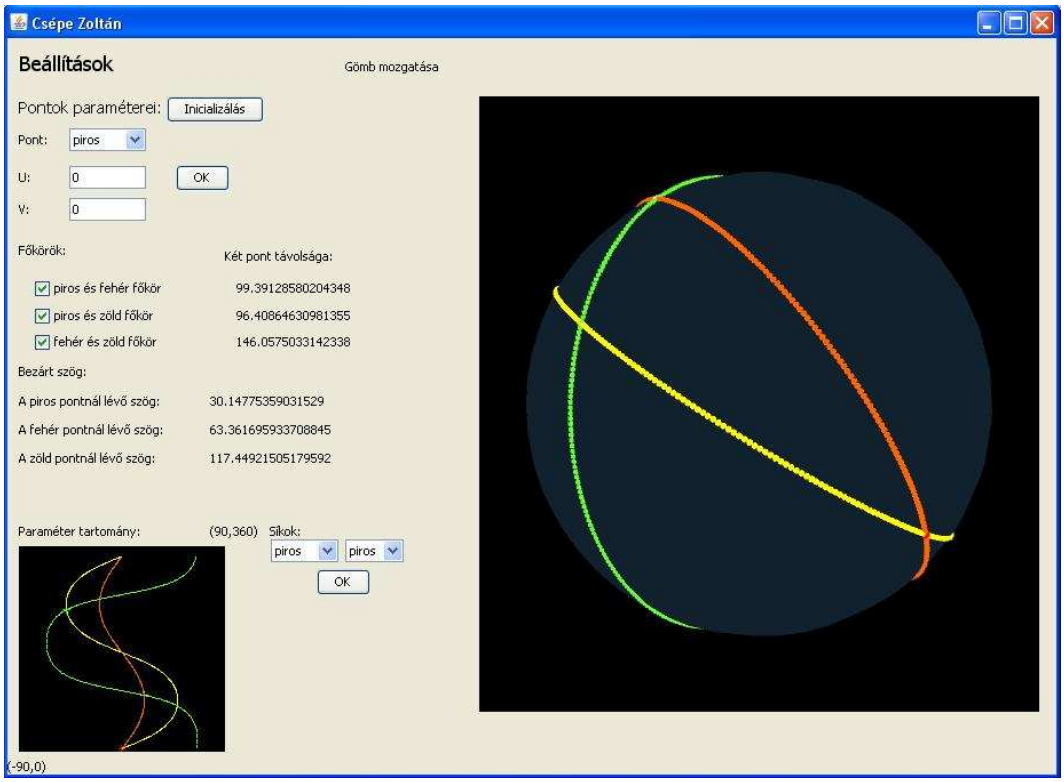
8. ábra A program indítás után

Látható, hogy két fő komponensre bonható a felhasználói interfész. Egyik a baloldalon látható beállításokat tartalmazó rész. Ahol a program által kezelt 3 pont paramétereit tudjuk beállítani, és egyéb dolgokat tudunk beállítani. A pontoknak az egyszerűség és a könnyebb kezelhetőség miatt a színük a nevük, így van piros, fehér és zöld pont. A

megjeleníteni kívánt főköröket és a hozzájuk tartozó síkokat is itt tudjuk be- és kikapcsolni. Továbbá itt írja ki a pontpárok távolságát és a három pont által meghatározott gömbháromszög csúcsainál lévő szögek nagyságát. Ha pontok *u* és *v* paramétereit szeretnénk megváltoztatni akkor a legördülő menüből kiválasztjuk a pont nevét majd egész értékeket megadva az OK gombra kattintva tudjuk rögzíteni az adatokat. Az *u* érték 0 és 360 közötti lehet, a *v* érték -90 és 90 közötti. Ekkor dinamikusan frissülnek a távolságok és a szögek. A síkok megjelenítésénél két legördülő menüből tudjuk kiválasztani a két pontot melyre illeszteni szeretnénk, majd az OK gombra kell kattintani. Ekkor ellenőrzi a program, hogy különböző pontokat adtunk-e meg, ha igen akkor megjelenik a sík, ha nem akkor a 9. ábraán lévő hiba üzenetet kapjuk.

9. ábra Hibaüzenet

Ha pedig a síkok által meghatározott főköröket szeretnénk megjeleníteni akkor a megfelelő jelölő négyzeteket kell bekapcsolni. Lehetőség van még az Inicializálás gombbal előre beállított adatokkal feltölteni a pontok paramétereit. Ekkor a piros (*u*,*v*) paramétere (0,0) lesz a fehér ponté(20,100) a zöld ponté (-50,260). A felhasználói interfész másik nagy egysége a megjelenítésre szolgál. Két panelből épül fel, a nagyobban jelenik meg a gömb és rajta a pontok, a főkörök és a síkok. Ezen a panelen lehetőség van az egérrel forgatni a gömböt. Valamint a scroll-lal közelíteni és távolítani. Illetve ha kétszer kattintunk az egér bal gombjával, akkor a nézőpont visszaáll az eredeti helyére. A kisebbik panelen a pontok paraméterei jelennek meg egy koordináta rendszerben, minden pont a nevének megfelelő színnel. Valamint a főkörökhöz tartozó paraméter görbék, melyek az őket meghatározó két pont színének keveréséből előálló színűek. Itt nincs lehetőség az egérrel való interakcióra. A program egy inicializálás utáni mikor minden főkört bekapcsoltunk állapotát a 10. ábra mutatja.



10. ábra A program futás közben

6.2. A szoftver készítésének lépései

Első lépésként a Swing elemeket elhelyeztem az ablakban. Majd ezután kezdetem az OpenGL-s rész megvalósításához. Ekkor kellet a gömböt a pontokat, a főköröket, síkokat és a paraméter görbéket megvalósítani.

6.2.1. A gömb létrehozása.

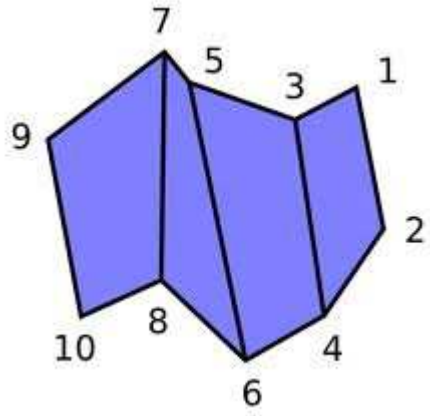
Első lépésként el kellett készítenem egy gömböt melyre rajzolni tudunk majd. Ehhez a gömböt szélességi és hosszúsági körökre bontottam, melyeknek a metszéspontjait meghatároztam a gömb paraméterezése alapján (11. ábra).

$$x = \cos(u) * \cos(v) * r$$
$$y = \cos(u) * \sin(v) * r$$
$$z = \sin(u) * r$$

11. ábra Gömb paraméterezése

Ezek alapján kis négyszög lapokat helyeztem el, melyek az OpenGL-be beépített elsimítás segítségével megadták a gömb felszínét. A gömbrajzoló függvény paraméter tartományában megadható a szélességi és hosszúsági körök száma és gömb sugara. A négyszöglapoknak rajzolásánál a GL_QUAD_STRIP-et használtam, melynek lényege, hogy meghatározunk két pontot, amelyek adják az első négyszög két csúcsát. Majd

ismét meghatározunk két pontot, amely a négyszög másik két csúcsát adják, de eközben a következő négyszög első két csúcsát is ezek adják. Ezután ugyanígy adjuk a négyszög hálózat többi pontját is. Így egy folytonos felületet kapunk (12. ábra).



12. ábra GL_QUAD_STRIP

Forrás: OpenGL tutorial – Lesson 21

6.2.2. A pontok létrehozása

A gömbre a pontokat a 8. ábrán látható gömbparaméterezés szerint helyeztem el. Mivel ilyenkor az OpenGL csak egyetlen pixelt helyez el, így valójában egy kis gömböt rajzoltam ki melynek a középpontja a pont helye a gömbön.

```
Pont(GL gl,int u, int v){
double a=Math.cos(u*jav)*Math.cos(v*jav)*20;
double b=Math.cos(u*jav)*Math.sin(v*jav)*20;
double c=Math.sin(u*jav)*20;
gl.glTranslated(a, b, c);
gomb(gl,0.3,5,5);
gl.glTranslated(-a, -b, -c);
}
```

A függvény megkapja paraméternek a rajzolás helyét és a pont (*u*,*v*) paramétereit. Ezekből kiszámolja pont koordinátáit a gömbön. Majd eltolja erre a helyre a rajzolás helyét, kirajzol egy 0,3 egységnyi sugarú gömböt. Majd visszatolja a rajzolás helyét az origóba.

6.2.3. A síkok létrehozása

A síkok valójában négyzetek melyeket a hozzájuk tartozó pontok és az origó segítségével határoztam meg. A síkokat rajzoló függvény paramétertartományában megkapja, hogy hol történik a rajzolás és a két pont (*u*,*v*) paramétereit. Először két

vektort határoz meg melyek az origóból a pontokba mutatnak. Majd definiálja a *z* egységvektort, hogy a *xy* síkban elhelyezett négyzetet majd be tudjuk forgatni a megfelelő helyre. Ezután kiszámolja a függvény a két ponthoz tartozó vektorok vektoriális szorzatát és normalizálja, mely nem más, mint a pontokra és az origóra illeszkedő sík normálvektora. Ezután meg kell határozni a forgatás tengelyét, amely a most kiszámolt normálvektor és *z* egységvektor vektoriális szorzata. A forgatásszögét pedig e két vektor skaláris szorzatából számolja ki a függvény. Ez után elforgatja a *xy* síkba rajzolt négyzetet és a megfelelő méretűvé skálázza. A végén egy ellentétes irányú forgatást végez, hogy a rajzolás helye visszakerüljön az eredeti állapotba.

```
Sik(GL gl, int u1, int v1, int u2, int v2){
    vektor a = new vektor();
    vektor b = new vektor();
    a.x=Math.cos(u1*jav)*Math.cos(v1*jav)*20;
    a.y=Math.cos(u1*jav)*Math.sin(v1*jav)*20;
    a.z=Math.sin(u1*jav)*20;
    b.x=Math.cos(u2*jav)*Math.cos(v2*jav)*20;
    b.y=Math.cos(u2*jav)*Math.sin(v2*jav)*20;
    b.z=Math.sin(u2*jav)*20;
    double s,szog;
    vektor ft,zt = new vektor();
    zt.x=0.0;
    zt.y=0.0;
    zt.z=1.0;
    vektor sn=a.vektorialisSzorzat(a,b);
    s=a.Norma(sn);
    sn.x=sn.x/s;
    sn.y=sn.y/s;
    sn.z=sn.z/s;
    s=a.Norma(zt);
    ft=a.vektorialisSzorzat(sn,zt);
    szog=-Math.acos(a.skalarisSzorzat(zt,sn))*180/Math.PI;
    gl.glRotated(szog, ft.x, ft.y, ft.z);
    gl.glScalef(30, 30, 30);
    gl.glBegin(gl.GL_QUADS);
    gl.glVertex3f(1,1,0);
    gl.glVertex3f(-1,1,0);
```

```
        gl.glVertex3f(-1,-1,0);
        gl.glVertex3f(1,-1,0);
        gl.glEnd();
        gl.glRotated(-szog, ft.x, ft.y, ft.z);
        gl.glScalef(0, 0, 0);
    }
```

6.2.4. A főkörök létrehozása

Tejesen azonos a síkok létrehozásával csak az *xy* síkban elhelyezett négyzet helyett egy *xy* síkban elhelyezett főkört forgat be. A *xy* síkban elhelyezett főkör meghatározása pedig kör paraméterezése alapján történik.

```
for(int i=0; i<360; i++){
            kor[i][0]=20*Math.cos(i*Math.PI/180);
            kor[i][1]=20*Math.sin(i*Math.PI/180);
            kor[i][2]=0.0;
        }
```

6.2.5. A paramétergörbe meghatározása

Adott a gömbön két pont és a rájuk illeszkedő főkör amely valójában egy felületi görbe. Azaz rögzített *v* érték mellett az *u* változtatásával kapunk meg a paraméterezés segítségével. Most a görbe pontjainak ismeretében kell a paraméter párokat meghatározni, melyek egy görbét határoznak meg a paraméter tartományban.

Itt is miként a síkoknál és főköröknél ismertetett módon meghatározzuk a forgatásszögét és tengelyét. Majd az ehhez tartozó transzformációs mátrixot letároljuk az m2 változóba. Ezután pedig a xy síkban elhelyezett főkör pontjainak koordinátáiból és a transzformációs mátrixból kiszámoljuk a beforgatott főkör pontjainak koordinátáit. Mivel az OpenGL homogén koordinátákkal dolgozik, ezért van szükség az s4-gyel való leosztásra.

```
gl.glRotated(-szog, ft.x, ft.y, ft.z);
        gl.glGetDoublev(gl.GL_MODELVIEW_MATRIX, m2, 0);
        gl.glRotated(szog, ft.x, ft.y, ft.z);
        for (int i=0; i<360; i++){
            s1=gkor[i][0]*m2[0]+gkor[i][1]*m2[1]+gkor[i][2]*m2[2]+m2[3];
            s2=gkor[i][0]*m2[4]+gkor[i][1]*m2[5]+gkor[i][2]*m2[6]+m2[7];
            s3=gkor[i][0]*m2[8]+gkor[i][1]*m2[9]+gkor[i][2]*m2[10]+m2[11];
```

```
        s4=gkor[i][0]*m2[12]+gkor[i][1]*m2[13]+gkor[i][2]*m2[14]+m2[15];
        parameterGorbe[a1][a2][i][0]=s1/s4;
        parameterGorbe[a1][a2][i][1]=s2/s4;
        parameterGorbe[a1][a2][i][2]=s3/s4; }
```

Ezután a megfelelő helyre került főkör pontjaiból ki kell számolni a hozzá tartozó paraméter görbe pontjait. Ekkor is a gömb paraméterezését (8. ábra) hívjuk segítségül. Továbbá tudjuk, hogy $x^2+y^2+z^2=1$ és $-1 \le z \le 1$. Egy pont z koordinátájából ki tudjuk számolni az u értéket, $u = \arcsin(z)$ ahol $-\frac{\pi}{2} < u < \frac{\pi}{2}$. Ezután kiszámolunk két segédértéket ezek $x_0 = \frac{x}{\cos(u)}$ és $y_0 = \frac{y}{\cos(u)}$, melyekre $x_0^2 + y_0^2 = 1$. Mivel

$$\frac{x^2}{\cos^2(u)} + \frac{y^2}{\cos^2(u)} = \frac{x^2+y^2}{\cos^2(u)} = \frac{1-z^2}{\cos^2(u)} = \frac{1-\sin^2(u)}{\cos^2(u)} = \frac{\cos^2(u)+\sin^2(u)-\sin^2(u)}{\cos^2(u)} = 1.$$

Így a v_0 értéke az x_0 értékétől függ, mely, ha negatív akkor a $v_0 = \pi - \arcsin(y_0)$ ha pedig x_0 nulla vagy annál nagyobb akkor $v_0 = \arcsin(y_0)$. Ekkor $-\frac{\pi}{2} \le v_0 \le \frac{3\pi}{2}$, tehát ha a v_0 értéke nulla vagy annál nagyobb akkor a $v=v_0$ egyébként pedig $v = 2\pi + v_0$. Ezután a megfelelő helyre toljuk a pixelt a kis rajzoló panelen és kirajzoljuk a görbét. Ahol a panel szélei folytonosak tehát ami véget ér a bal oldalon az folytatódik a jobbon, és ugyanígy a felső és alsó szélen is.

```
    gl.glBegin(gl.GL_POINTS);
    for(int i=0; i<360; i++){
      su=Math.toDegrees(Math.asin(parameterGorbe[a1][a2][i][2]));
      x0=parameterGorbe[a1][a2][i][0]/Math.cos(su*Math.PI/180);
      y0=parameterGorbe[a1][a2][i][1]/Math.cos(su*Math.PI/180);
      if (x0>0) sv0=Math.toDegrees(Math.asin(y0));
      if (x0<=0) sv0=180-Math.toDegrees(Math.asin(y0));
      if (sv0<0) sv=360+sv0;
      else
          sv = sv0;
      su1= ((su%360)*0.31);
      sv1= (((sv%360)*0.15)-28.0);
      gl.glVertex3d(su1, sv1, 0);
    }
    gl.glEnd();
```

6.2.6. A vektor osztály

A 7.2.3. fejezetben is látható a síkok meghatározásánál szerepel egy vektor osztály. Ez egy általam definiált osztály, mely egy origóból induló vektor reprezentálására képes és vektor műveleteket tudunk végezni. Az osztály elemei:

- Vektor – (*x*,*y*,*z*)koordinátákkal rendelkező háromdimenziós vektor;
- Vektoriális szorzat – két vektor vektoriális szorzatának eredménye egy harmadik vektor mely merőleges mindkettőre és velük jobb sodrásos rendszer alkot és $|a \times b| = |a| \cdot |b| \cdot \sin(\varphi)$ ahol φ a két vektor által bezárt szög;
- Norma – egy vektor nagyságát adja meg, $|a| = \sqrt{x^2 + y^2 + z^2}$;
- Skaláris szorzat – két vektorhoz egy számot rendel a következő módon $a \cdot b = |a| \cdot |b| \cdot \cos(\varphi)$ ahol φ a két vektor által bezárt szög.

```
vektor vektorialisSzorzat(vektor a, vektor b){
        vektor c= new vektor();
        c.x=a.y*b.z-b.y*a.z;
        c.y=-(a.x*b.z-b.x*a.z);
        c.z=a.x*b.y-b.x*a.y;
        return c;
    }
```

7. Néhány feladat megoldása

A fejezetben felhasznált irodalom:

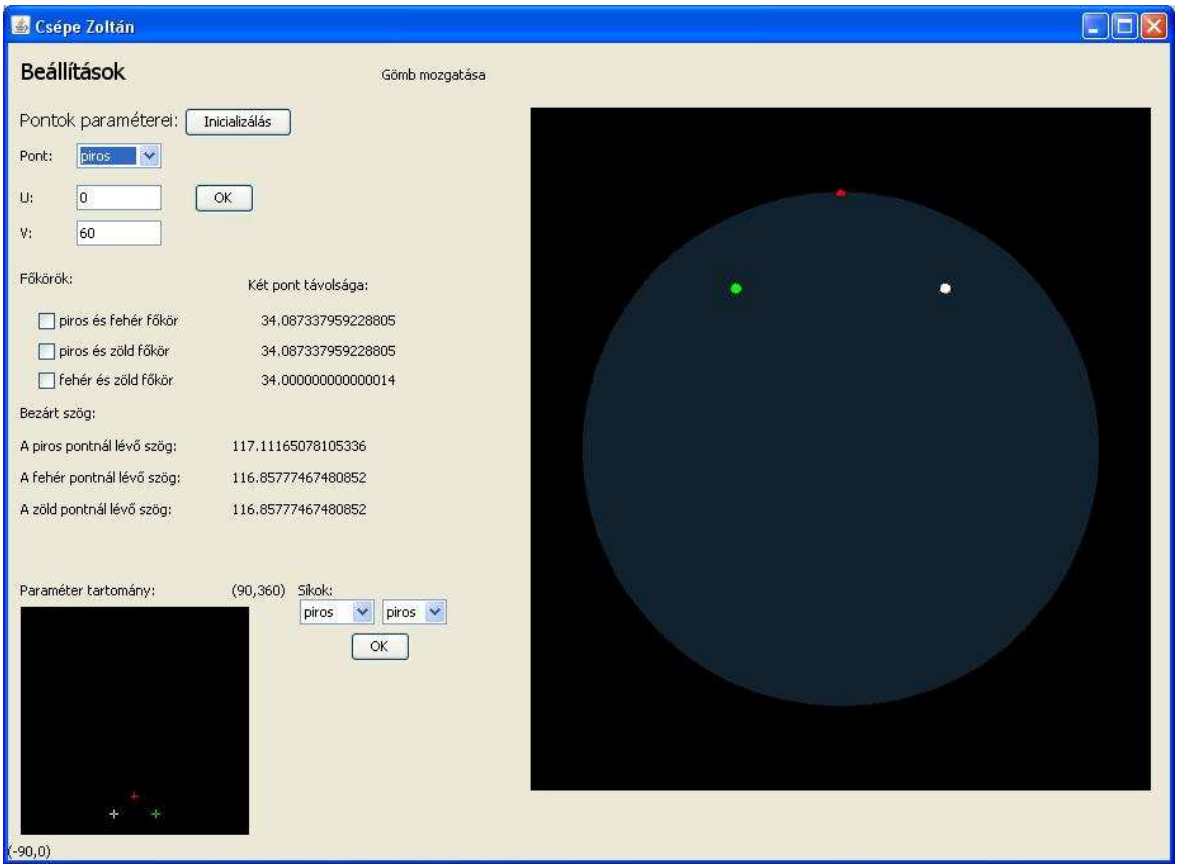
- LÉNÁRT I.: Sík és gömb, Múzsák kiadó
- JOEL CASTELLANOS: Mi is a nem euklideszi geometria?
- Vígh-Kiss Erika személyes honlapja

A következőkben szeretnék bemutatni néhány feladatot, a Lénárt féle munkalapból a program segítségével meg lehet oldani.

7.1. Milyen színű a medve?

0.1. Kísérlet: Medve úr elindul hazulról és 34 kilométert vándorol dél felé. Pihen egy sort, azután megfordul nyugat felé, és megy egyenesen előre újabb 34 kilométert. Majd újra megfordul, és észak felé indul. Egy idő után meglepetten látja, hogy hazaért. Milyen színű a medve?

A feladat egy figyelemfelkeltő bevezető feladat. A gyerekek rögtön kísérletezéssel neki kezdhetnek a feladat megoldásának. Ez lehet a Lénárt gömbbel vagy a programmal. Ha a Lénárt gömbön próbálgatnak, akkor előbb utóbb eljutnak oda, hogy a medve a déli vagy az északi sarkon él. Ez azzal áll összefüggésben, hogy a kiinduló és végpontnak egybe kell esnie. A programmal is dolgozhatnak, mivel a program három pontot tud kezelni, így pont ideális is megoldáshoz. Először meghatározzuk az indulás helyét. A piros pont paramétere legyen a (*0,180*) a paramétertartomány közepe, innen könnyen tudunk, bármely irányba indulni. Menjünk délre 34 kilométert, és helyezzük el a fehér pontot ide. Ekkor a paraméterei (*0,146*). Innen pedig menjünk nyugatra 34 kilométert ekkor a zöld pont paraméterei (*-34,146*). Itt már látszódik a paramétertartomány ábráján, hogy hiába indulunk innen északnak nem fogunk vissza érni a kiindulási helyre. A gyerekek ekkor már sejthetik, hogy nem a síkon kell gondolkozni, hanem a gömbön. Ekkor rákérdezhetünk, hogy a földgömbön melyik pont lenne jó kiindulási pontnak, ez nem más, mint az északi pólus. A programban a bemutatáshoz, ha piros pontnak a (*0,60*), a fehérnek a (*-17,30*) és zöldnek a (*17,30*) paramétereket adjuk, és beforgatjuk, hogy a piros pont az északi sarkpont legyen, akkor rögtön látszódik itt is a megoldás. A pontok távolságánál pedig leolvasható a medve által megtett út is, amiből látszik, hogy a gömbháromszög szabályos (13. ábra). Tehát a medve fehérszínű.



13. ábra Milyen színű a medve

7.2. Alapfogalmak

Az 1.1. kísérletben a legegyszerűbb alakzatot keressük meg a gyerekekkel. Ez a program első indításakor rögtön adódik, hogy ez a pont. Mivel a programban nem tudunk semmilyen műveletet végezni, még a pontok (*u, v*) paramétereit nem adjuk meg.

Az 1.2. kísérletben a síkbeli egyenes gömbi megfelelőjét keressük. Ez is rögtön adódik miután elhelyeztünk két pontot a gömbön, már rögtön rájuk tudunk illeszteni egy főkört, és könnyen tudjuk ki és bekapcsolni. Valamint a pontok paramétereinek változtatásával tudjuk a főkör változását is vizsgálni.

Az 1.3. és 1.5. kísérletben a távolság- és szögméréseket ismerik meg a gyerekek. A programban ezek a pontok paramétereitől függően mindig dinamikusan frissülnek. Így könnyedén tudunk velünk számolni. Az egyes értékek fokokban írja ki a program. Melyeket akár össze is adhatunk szemben a síkkal.

Az 1.4. es paraméterben a sarkpontok megszerkesztése az egyik feladat. A programban könnyedén tudunk megadni ilyen pontokat. Az *u* vagy az *v* paraméterek különbsége száznyolcvannak kell lennie, ilyen például a (*0,0*) és a (*0,180*). A másik

feladat az egyenlítő szerkesztése, a (*-90,0*) sarkponthoz az előbbi két pont által meghatározott főkör az egyenlítő.

7.3. Párhuzamosok és merőlegesek

A 2.1. kísérletben a főkörök metszéspontjának számát kell meghatározni. A programba eddig bevitt példákon is jól látszik, hogy a síkbeli egyenesekkel szemben két főkörnek, ha nem esnek egybe, akkor két metszés pontja van. Továbbá az is látszik, hogy mindig van két főkörnek legalább két közös pontja így nem tudunk párhuzamos főköröket megadni. Itt mutatkozik az egyik nagy különbség az euklideszi geometriával, a párhuzamossági axióma nem áll fenn a gömbi geometriában.

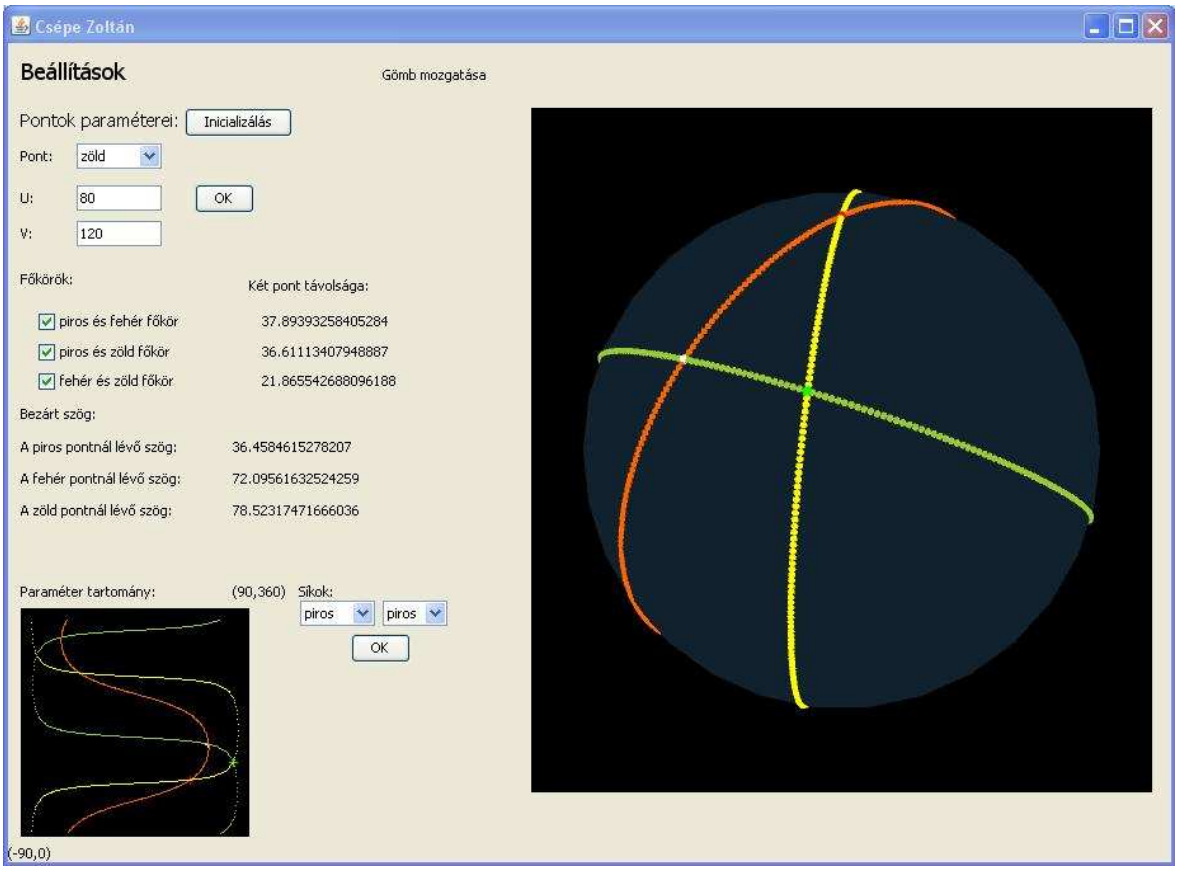
A 2.2.-es és a 2.3.-as kísérletben merőleges főkörök helyzetét kell vizsgálni. Két merőleges főkört határoznak meg a (*0,0*) (*90,0*) és a (*0,180*) paraméterek. Melyek jól szemléltetik, hogy két merőleges főkör két pontban metszi egymást, nyolc derékszöget határoz meg és négy egybevágó véges részre osztja a gömbfelületet. Szemben a síkbeli esettel ahol két merőleges egyenesnek egy közös pontja van, négy derékszöget határoznak meg és a síkot négy végtelen egybevágó részre bontja szét. A másik érdekesség, amit vizsgálnunk kell, hogy két főkörnek hány közös merőlegese van. Ha síkban nézzük, akkor két metsző egyenesnek nincs közös merőlegese, két párhuzamos egyenesnek pedig végtelen sok közös merőleges van. Ezzel szemben két főkörnek mindig van egyetlen közös merőlegese és két merőleges főkörnek is csak egy merőleges főköre van. Ezen példa bemutatása a (*0,0*), a (*0,90*) és a (*90,0*) paraméterek által meghatározott pontok alkalmasak.

7.4. Sokszögek

A 3.1. kísérletben arra keressük a választ, hogy van-e olyan sokszög, amelynek két oldala van. A síkon nem lehetséges ilyen objektum mivel két közös pontból induló félegyenesnek nincs további közös pontjuk és síkot két végtelen síktartományra bontja. Ezzel szemben a két közös pontból kiinduló főkör ismét metszeni fogják egymást az át ellenes pontban. Ekkor keletkezik a gömbkétszög. Ennek a bemutatására el kell helyezni a programban három pontot erre akár az Inicializálás gombot is használhatjuk. Ezután két tetszőleges főkört bekapcsolunk, és rögtön látható a gömbkétszög.

A 3.2.-es és a 3.3.-as kísérletek mind a gömbi háromszöghöz kapcsolódnak. Ezekhez a feladatokhoz mind a három pontra paramétereit meg kell adni. Például (*45,90*), (*60,150*), (*80,120*) megfelelő értékek, ekkor látszódni fog, hogy három pont és a rájuk

illeszkedő három főkör nyolc gömbháromszöget határoz meg és a gömb felszínét nyolc véges tartományra bontja (14. ábra).



14. ábra Gömbi háromszög a programban

A 3.4.-es és a 3.5.-ös kísérletek a gömbháromszög szögeire vonatkoznak. Már az eddigi feladatokban is láttuk, hogy a gömbön előfordulhat olyan eset mikor a gömbháromszög szögeinek összege nagyobb, mint 180 fok. Az a kérdés hogy mi a maximális érték, amit elérhet a gömbháromszög szögeinek összege, ez nem más, mint az 540 fok, amit akkor kapunk, ha a három pont egy főkörre esik. Ekkor ez valójában egy elfajuló gömbháromszög. De csinálhatunk belőle nem-elfajuló gömbháromszöget, ha az egyik csúcsot elmozdítom ε-nal. Ekkor a szögösszeg 540-δ. Tehát nem-elfajuló gömbháromszöggel is tetszőlegesen közel kerülhetünk az 540 fokhoz. A másik kérdés hogy lehet-e három derékszöge egy gömbháromszögnek, erre láttunk már példát 8.3. fejezetben mikor két merőleges főkörre merőleges főkört kerestünk.

8. Egy tanítási óra a program használatával

A következőkben egy tanítási óra vázlatát szeretném bemutatni. Melyen a Lénárt-gömb mellett az én programomat is használnák a gyerekek. Az órát számítógépes terembe terveztem ahol minden gyerek saját maga használja a programot. Három gyerekenként jutna egy rajzgömb készlet, a teremben szükség lenne egy projektorra és egy tanári számítógépre a könnyű szemlélhetőség kedvéért.

Idő	Az óra menete	Megjegyzések
0-10. perc	Megbeszéljük, hogy a síkon a legegyszerűbb elem a pont. Majd ennek mintájára megállapítjuk, hogy a gömbön is a legegyszerűbb alakzat a pont. Rajzoljunk a Lénárt-gömbre két pontot és a programban is helyezzünk el két pontot. A gyerekeknek itt tudjuk megmutatni a kapcsolatot a sík és a gömb között. Felírjuk a táblára a gömb paraméterezését és néhány (u,v) paraméterpárra számológép segítségével meghatározzuk, melyik pont tartozik a gömbön hozzá (15. ábra).	Az előző órán már foglalkoztunk a gömbi geometriával. A kedvcsináló medvés feladatot átbeszéltük. Az óra e részében kapcsoljuk be a számító gépet és indítjuk el a programot.
10-20. perc	Ha elhelyeztünk már egy pontot a gömbön helyezzünk el még egyet. Vizsgáljuk meg, hogy ha a síkon adott két pont, akkor arra milyen objektumot tudunk illeszteni. A válasz természetesen az egyenes. Ezután megvizsgáljuk, hogy a gömbön két adott pontra tudunk-e egyenest illeszteni. Ekkor a gyerekek megismerkednek a főkör fogalmával. A Lénárt gömbön a gömbi vonalzó segítségével megrajzolják azt, míg a programban bekapcsolják a főkör megjelenítését. **Feladat**: Gyűjtsd össze a sík egyenes vonalával és a gömb főkörével kapcsolatos megfigyeléseidet.	A feladat alapja a 1.2 kísérlet 13. feladata. Csoportmunkában először a gyerekek összeszedik, a tulajdonságokat majd közösen megvitatjuk őket.

	Megoldás: *Egyenes* - Az egyenes vonal végtelen, nincs középpontja. Ha követünk egy egyenes vonal sosem érünk vissza a kiindulási pontba. Két ponton át mindig egy egyenes húzható. Két pont az egyenes három részre bontja, egy véges és két végtelen. *Főkör* - A főkör véges. Ha követünk egy főkört, akkor visszaérünk a kiindulási pontba. Két főkörön általában egy főkör húzható, de van olyan speciális helyzete a két pontnak hogy végtelen sok főkör húzható rajtuk. Két pont a főkört két véges részre bontja.	
21-24. perc	Ezután helyezzünk el még egy pontot a gömbön. Majd bekapcsoljuk a főköröket, a programban, a Lénárt-gömbön pedig megrajzoljuk a főköröket. Ekkor a síkbeli háromszög mintájára létrejön a gömbháromszög.	A gyerekek a pont bevitele után kicsit forgatják, tanulmányozzák gömböt és rajta a pontokat és a főköröket. Érdemes a pontokat úgy megadni, hogy egyszerre látszódjon mind a három. (14. ábra)
25-27.perc	**Feladat**: Hány háromszög jött létre a gömbön? Megoldás: 8	A gyerekek önállóan megszámolják a háromszögeket a program segítségével, majd jelentkezés útján megmondhatják a választ.
28-35. perc	**Feladat**: Lehet-e egy gömbháromszögnek két derékszöge	Mindenki önállóan dolgozik, majd

	és három? Megoldás: Igen A gyerekek a gömbháromszög csúcsainak paramétereit változtatják a programban vagy átrajzolják a Lénárt-gömbön a meglévő ábrájukat.	megbeszéljük a választ és bemutatok egy példát a megfelelő paraméterezésre.
35-40. perc	Most pedig vizsgáljuk meg a gömbháromszög belső szögeinek összegét. A síkban már ismerjük, hogy bármely háromszög belső szögeinek összege 180 fok. Ezzel szemben a gömbön ez nem igaz. Az előző feladatban is láttuk, hogy ha már csak két derékszöge van, akkor is a szögeinek összege nagyobb, mint 180 fok. Keresnünk kellene egy számot, amelynél nagyobb már nem lehet. Ez nem más, mint az 540 fok, amely egy elfajult gömbháromszög és egy főkörnek látszik.	A gyerekekkel közösen próbálgatással jutunk el a megoldásig
40-45. perc	**Összefoglalás**: A mai órán megismerkedtünk a főkörrel és a gömbi háromszöggel. Valamint megtanultunk egy nagyon fontos különbséget a síkbeli és gömbháromszögek között. **Házi feladat 1**: Gondolkozzatok el azon, hogy van-e a gömbháromszögnél egyszerűbb alakzat a gömbön. Megoldás: Igen, a gömbkétszög. **Házi feladat 2**: Keressetek olyan városokat a földgömbön amelyek egy főkörön vannak! Megoldás: Alexandria – Siena Kanton – Dhaka	

	London - Accra	

8.1. Táblaképek

A gömbi geometria

Paraméterezés:

$$x = \cos(u) * \cos(v) * r$$

$$y = \cos(u) * \sin(v) * r$$

$$z = \sin(u) * r$$

Példák:

r=20

(*u*,*v*) -> (*x*,*y*,*z*)

(0,90) -> (0,20,0)

(30,45) -> (12,25;12,25;10)

15. ábra 1. táblakép

9. Összefoglalás

A dolgozatomban bemutattam egy a középiskolai tananyagon túlmutató geometria ágazat megértését segítő program megvalósítását és használatát. Mely felhasználóbarát kezelőfelülettel rendelkezik, hardvertámogatással képes grafikus tartalmat megjeleníteni, jelenestben gömbi geometria alap elemeit. Egy eszköz együttest sikerült kialakítanom melyekkel könnyedén lehetett a fejlesztést megvalósítani. A következő eszközöket használtam:

- OpenGL API
- Java Swing
- JOGL
- NetBeans fejlesztői környezet

A program a gömbi geometria alapjainak megértését segíti. Lehetőség van a paraméter tartományon három darab pontot elhelyezni melyeknek megfelelő pontok a gömb felszínén is meg jelennek. A pont párokra illeszkedő síkokat legördülő menük és gombok segítségével lehet bekapcsolni, míg a főköröket jelölőnégyzetek segítségével. A program a pontok távolságát és a főkörök által bezárt szögeket dinamikusan mindig az aktuális paramétereknek megfelelően írja ki. A kényelmes használat érdekében a gömböt egérrel lehet mozgatni, de mindig vissza tudunk térni az eredeti helyzetbe. A megvalósításnál fontos volt a look&feel módszer megfelelő alkalmazása, hogy a használat intuitív módon történjen. A gömb és a pontok paraméter tartományról való leképezését a gömb paraméterezésével valósítottam meg. A főkörök és síkok meghatározásánál a két adott pontra és az origóra illeszkedő síkot határoztam meg, majd az xy síkben elhelyezkedő megfelelő objektumot forgattam be.

A program még nem teljesen valósítja meg rajzgömb készlet elemeit, ez későbbi megvalósításra vár. A pontok tetszőleges számának bevitel fontos lenne, valamint a gömbi körző elkészítés. De a rajzgömb készlethez képest többlet szolgáltatása is van, ez a főkörökhöz tartozó paraméter görbék kirajzolása.

10. Ábrajegyzék

11. Irodalomjegyzék

[1] KÁLMÁN A.: Nemeuklideszi geometriák elemei, Nemzeti Tankönyvkiadó 2002

[2] LÉNÁRT I.: Sík és gömb, Múzsák kiadó

[3] G. HORVÁTH Á. – SZIRMAI J.: Nemeuklideszi geometriák modelljei, Typotex kiadó 2004

[4] FEKETE Á. ZS.: Másodrendű felületek ábrázolása, Debreceni Egyetem szakdolgozat 2010

[5] KUBA A.: Az OpenGL grafikai rendszer, Egyetemi jegyzet http://www.inf.u-szeged.hu/oktatas/jegyzetek/KubaAttila/opengl_html/

Internetes források:

[6] JOEL CASTELLANOS.: Mi is a nem euklideszi geometria? http://www.ngkszki.hu/~trembe/noneuclid/hungarian/noneuclidean.html

[7] NetBeans leírás: http://netbeans.org/index_hu.html

[8] OpenGL tutorial – Lesson 21 http://www.videotutorialsrock.com/opengl_tutorial/crab_pong/text.php

[9] NeHe Tutorial JOGL Port - Sphere mapping quadrics in OpenGL http://www.java-tips.org/other-api-tips/jogl/sphere-mapping-quadrics-in-opengl-nehe-tutorial-jogl-2.html

[10] Wikipedia – Java OpenGL http://en.wikipedia.org/wiki/Java_OpenGL

[11] KAI RUHL: JOGL Tutorial http://www.land-of-kain.de/docs/jogl/

[12] Vígh-Kiss Erika személyes honlapja http://www.gombigeometria.eoldal.hu/

Köszönetnyilvánítás

Köszönetemet szeretném kifejezni témavezetőmnek **Dr. Nagy Gábor Péter** docens úrnak a dolgozatom elkészítéséhez nyújtott szakmai segítségért, valamint, hogy lehetővé tette számomra az SZTE Bolyai Intézet Geometria Tanszékén a dolgozat megírását. Valamint köszönetemet szeretném kifejezni szüleimnek, családomnak, barátaimnak a támogatásért.

Nyilatkozat

Alulírott Csépe Zoltán informatikatanári-matematika szakos hallgató, kijelentem, hogy a diplomadolgozatban foglaltak saját munkám eredményei, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem azt, hogy szakdolgozatomat a Szegedi Tudományegyetem könyvtárában, a kölcsönözhető könyvek között helyezik el.

.....

Aláírás

Szeged, 2010. május 13.