MMA szofverfejlesztés és bitmap transzformációk

MMA Software Developing and Bitmap Transformations

doktori (PhD) értekezés tézisei

Tornai Róbert

Debreceni Egyetem
Debrecen, 2005.

MMA szofverfejlesztés és bitmap transzformációk

MMA Software Developing and Bitmap Transformations

doktori (PhD) értekezés tézisei

Tornai Róbert

Debreceni Egyetem
Debrecen, 2005.

1. MMA technológia

Az MMA (Multiplex Microbead Assay = mikrogyöngy alapú multiplex analitika) technológia egy analitikai módszer. Az MMA technológia legnagyobb előnye az, hogy egyetlen mérés során több reagenst használhatunk, ha különböző festékkel jelöljük meg a különböző reagensekkel fedett gyöngyöket. Ily módon a szükséges (vér, plazma, oldat vagy egyéb) minta mennyisége jelentősen kisebb lehet. Legnagyobb eredménynek azt tartom, hogy tanulmányozva az MMA technológiát használók szokásait, kialakult munkafolyamatait, sikerült kialakítani és dokumentálni egy olyan módszert, ami számítógépes programokkal támogatható. Ezen túl, elkészítettük az eredmények értékelésének automatizálását is, illetve az eredmények összeállítását, nyomtatását és azok hitelességi problémáit is kielégítő módon kezeltük.

1.1 Az MMA technológia alapjai

Egy festék különböző koncentrációban való használatával maximum 10-16 gyöngyöt különböztethetünk meg egy mérés során. Két csatorna adatainak használatával két festéket különböztethetünk meg, így a megkülönböztethető gyöngyök maximális száma 256-ra emelkedik. A megkülönböztethető gyöngyök maximális száma különböző méretű gyöngyök használatával terjeszthető ki.

A SAT Flow szoftver segítségével a kutatók megtervezhetik a kísérleteiket és az ASF Creator program segítségével kit-eket készíthetnek hozzájuk. Egy kit reagensek és oldatok készlete, ami az adott kísérlet típusához igazodik. A kit-ekhez mellékelt leíró állományt Assay Specification fájlnak (rövidítve: AS fájlnak vagy ASF-nek nevezzük).

Az eredeti elképzelés szerint két program készült. Egy (ASF Creator) az AS fájlok létrehozásához és egy (SAT Flow) a kísérletek tervezéséhez és kiértékeléséhez. Ez két különböző felhasználói csoportot jelentett. Csak a kit összeállítója volt jogosult módosításokat eszközölni az AS fájlokon. Ennélfogva az ASF Creator program egy titkosító algoritmussal védett. Később az elvárások megváltozásával a Soft Flow Kft. (lásd [10]) a programok egyesítése mellett döntött. Az elkészült programok nagymértékben megkönnyítik a biológiai témákban kutatók kísérleteinek tervezését és kiértékelését.

1.2 Egy mérés lépései

Egy mérés a következő lépésekből áll:

- Először is a mintát össze kell keverni a megfelelő reagensekkel.
- Az így nyert oldatot szerves fluoreszkáló festékkel kell elkeverni, ami felkötődik a reagensekre.
- Az oldatot a készülékbe kell helyezni.
- A felhígított oldat egy kapillárison keresztül a cuvet-be kerül.
- Lézer világítja meg a nagysebességű oldatot. A lézerfény több irányban is visszaverődik, illetve elnyelődik. Elnyelődés esetén a festékmolekula fényt kezd kibocsátani.
- Az 'esemény' kifejezést a lézernyaláb előtt elhaladó mintára használjuk.
- A visszaverődő fény intenzitását megfelelően elhelyezett szenzorokkal rögzítjük.
- A jeleket a cytometer feldolgozó egysége digitalizálja. A régebbi készülékek 8, 10, 15, 16 bitet használnak, a legújabb modellek pedig 32 bitet. Általában lehetőség van a jelek logaritmikus konverziójára a jobb eredmények érdekében.
- A digitalizált csatorna értékekből a gyűjtő számítógépben készül egy List Mode file (rövidített alakja: LM file) az egyes mérésekhez.

1.3 Az ASF Creator szoftver

Egy új kit létrehozása az összes adat megadásával vagy egy létező kit adatainak a módosításával lehetséges. Egy kit vagy kvantitatív vagy qualitatív kiértékelési modellt támogat.

A kit összeállítója egy nevet, azonosítót rendel a kit-hez. A LOT egy egyedi szám, ami egy sarzsot azonosít. A név - azonosító - LOT hármas egyértelműen azonosít egy kit-et. A klaszterek száma megadja azt, hogy az AS file mennyi klasztert fog tartalmazni, amiket a klaszterező paraméterek határoznak majd meg. A lejárati idő a készítési idő plusz az alapértelmezett szavatossági idő.

Csak azon készülékek LM file-jai lesznek használhatóak a SAT Flow szoftverben, melyek a megfelelő paraméter sorrenddel fel vannak sorolva itt. Újabb készülékek a velük készült LM file-ok segítségével adhatóak meg.

Következő lépés a kit adatait tartalmazó LM (List Mode) file megadása, amit az ismert készülékek egyikével kell elkészíteni. A paraméterek a készüléklista alapján felveszik az alapértéküket.

Ha az automatikus klaszterezés során nem megfelelő számú klasztert talál a program, akkor a felhasználónak ellenőriznie kell a paramétereket. Sikeres klaszterezés esetén az analyte-okat párba kell rendezni a gyöngyökkel.

1.3.1 Kvantitatív modell

A varázsló oldalak eddig a lépésig a kvantitatív és a kvalitatív modellekben megegyeznek.

A kvantitatív modellben minden egyes analyte-hoz egy függvényt kell rendelni, például a harmadfokú, a négyparaméteres logisztikus görbe és az exponenciális szigmoid függvény választható ki.

A standard-ek adják az alappontjait annak a görbének, ami az intenzitás értékeket konvertálja koncentráció értékekké. A kvantitatív modell illesztő függvényei a paramétereket a mért intenzitás értékekből és a megadott standard koncentrációkból nyerik, amelyekre görbét illesztenek. A tesztminták intenzitás értékeit behelyettesítve a megfelelő függvénybe a koncentráció értéket kapjuk. Egy kit tetszőleges számú standard-et tartalmazhat, azonban a költségek miatt a standard minták általában ésszerű mértékben vannak jelen.

A kvantitatív kontrollmintáknak három koncentrációja van: alacsony, közepes és nagy. A Low C, Medium C és High C oszlopok tartalmazzák a névleges koncentráció értékeket az analyte-okhoz. A Low dx, Medium dx és High dx oszlopok tartalmazzák a megengedett eltérést a névleges koncentrációktól százalékban kifejezve. Az üzenet oszlop a SAT Flow program riportjaiban megjelenő figyelmeztető szövegeket tartalmazza a koncentrációk intervallumtól való eltéréséhez. A Number of replicates mező a kontrollminták ismétlési számát tartalmazza.

Ha egy analyte koncentrációja kielégít bizonyos feltételeket, akkor a felhasználó szöveges üzenetet kap, például egy diagnózist. Ezek az üzenetek a következő varázsló lapon adhatóak meg.

A varázsló utolsó előtti oldalán megadható, hogy mennyi készülékkalibráló gyöngy legyen a kit-ben. Meg lehet megadni, hogy a készülékkalibráló gyöngykeveréknek milyen legyen az összetétele. A gyöngyök Relatív Fluoreszcens Intenzitás (RFI) értéke egyenként megadható. Ezek a standard-ek tetszőleges számban lehetnek jelen, csak a költség szab határokat.

Az utolsó információs oldal összegzi az eddigi döntéseink eredményét, amit a felhasználó kinyomtathat és file-ba menthet.

Mentés után a Save... gomb kiszürkül, mivel egy kit csak egyszer menthető el. A Finish gomb megnyomásával lehet befejezni a munkát, ha a kit nincs kinyomtatva, akkor a felhasználó figyelmeztetést kap. Ugyan a nyomtatás nem kötelező, de erősen ajánlott.

1.3.2 Kvalitatív modell

A kvalitatív modellben a következő oldal a modell adatainak megadására szolgál. A pozitív és negatív kontrollminták egymástól függetlenül definiálhatóak. Gyakran csak az egyik fajta minta van jelen a kit-ben.

A következő két lépés megegyezik a kvantitatív ággal. A felhasználó megadja az analyte-okhoz tartozó üzeneteket és a készülékhez tartozó standard gyöngyöket. A legfontosabb különbség, hogy ezek az üzenetek Cut-Off Indexeken alapulnak a koncentráció értékek helyett. A végső mentési illetve nyomtatási eljárás szintén megegyezik.

1.4 AS Files

Az ASF Creator program egy AS file-t készít, ami egy kit-et támogat. Tartalma a következő:

- A kiértékelés típusa (kvalitatív, kvantitatív, IS = Instrument Standard)
- Név, ami leírja a kit funkcióját
- A készítő által adott katalógus azonosító
- Sorozat azonosító (LOT szám)
- Egyedi azonosító szám
- A támogatott citométerek neve
- Karaktersorozat, ami egy citométert azonosít a List Mode file fejlécében
- Festék azonosító paraméterek
- Méret azonosító paraméter
- Riport paraméter
- Analyte lista, a klaszterezési eljárás által definiált sorrendben
- Gyöngy lista, a klaszterezési eljárás által definiált sorrendben
- Feltételes üzenetek típusai, határai és az üzenetek szövegei

Kvantitatív esetben:

- Az illesztendő függvény típusa

- A standard minták koncentrációi analyt-onként és a koncentráció mértékegysége

Kvalitatív esetben:

- Cut-Off Indexek (COI) a pozitív és negatív kontrollmintákhoz analyte-onként és a tűréshatár

Az Instrument Standard esetben:

- Nincsenek klaszterező paraméterek, festék lista, kontrollminták és feltételes üzenetek
- Relatív Fluoreszcens Intenzitás (RFI) érték minden egyes gyöngyhöz

1.5 A SAT Flow szoftver

Egy kísérlet lépései:

- A kutató eldönti a kísérlet típusát. Megadja a reagenseket és meghatározza, hogy kvalitatív vagy kvantitatív mérésre van-e szükség, a készüléhez tartozó standard mintákkal ellenőrizni tudja a citométer integritását.
- Az igényeknek megfelelő kit kiválasztása után importálni kell az AS file-t a SAT programba.
- Egy kit többféleképpen használható fel:
  - Kötegelt felhasználás: Egy adott tesztet kell elvégezni számos különböző forrásból származó mintán. Ezeket a mintákat kötegelt vizsgálattal lehet analizálni.
  - Mintaorientált felhasználás: Betegek esetén használható. Az egyes mintákhoz különböző vizsgálatokat kell rendelni. Ekkor az egy forrásból származó mintán számos tesztet lehet elvégezni.
- A kutató tesztmintákat adhat egy vizsgálathoz vagy módosíthatja is tesztet. A kísérő minták szükséges mennyisége az AS file-ban van letárolva. Ezek a nem módosítható minták a következők:
  - Standard-ek - alappontjai annak a görbének, ami kvantitatív esetben az intenzitás értékeket konvertálja koncentrációvá. Az alappont második koordinátáját az LM file-ban megadott ismert koncentrációjú standard minta intenzitás értéke szolgáltatja.

  - Kvantitatív kontroll - ellenőrzési célokat szolgál, amennyiben nincs egy tartományon belül, akkor a kiértékelési eljárás egy hiba üzenetet ad.
  - Kvalitatív kontroll - a kvalitatív kontrollmintáknak hasonló szerepe van mint a standard-eknek a kvantitatív modellben. A minták határozzák meg a függvény paramétereit, ami a Cut-Off Indexek kiszámításához szükséges.
  - Készülék standard - festék nélküli calibráló gyöngyöket tartalmaz. A készülékhez kapcsolódó standard-ek használata kikapcsolható.

- A minták egy mikrogyöngy tálcán vagy egy rácsos tartón helyezhetőek el. Egy tálca 96 vagy 384 helyet tartalmaz mátrixos elrendezésben. A jobb eredmények érdekében bármelyik mintához kérhető ismétlés, ami természetesen ugyanabból az oldatból származik. Ezekután egy felhasználási térkép nyomtatható a technikusok számára. Ez a térkép mutatja, hogy melyik oldatot melyik helyre kell helyezni.

- A technikusok végrehajtják a mérést.

- Az LM file-okat egy mappába kell másolni, hogy a Data File Assignment segítségével a mintákat automatikusan hozzárendelhessük a List Mode file-okhoz. A hozzárendelés kézzel módosítható szükség szerint.

- A riport formátumát és megjelenítési módját legkésőbb ezen a ponton kell eldöntenünk. Kétféle riport készíthető, egy rövidebb „Summary Report" és egy részletesebb „Full Report".

- Miután minden be van állítva, elkezdődhet a kiértékelési eljárás a List Mode file-ok feldolgozásával. Az AS file-okban rögzített paraméterek segítségével megtörténik a klaszterezés, ami az egy reagenshez tartozó események kiválogatását jelenti. A kvalitatív és kvantitatív módszerek eltérő kiértékelési eljárásokat alkalmaznak. Kvantitatív kiértékelés esetén egy görbeillesztésről van szó a standard minták alapján az AS file-ban rögzített görbetípus alkalmazásával. Kvalitatív kiértékelés esetén a pozitív és negatív kontrollminták intenzitásai alapján egy Cut-Off Indexet számolunk. A tesztminták intenzitás értékeit elosztjuk ezzel a Cut-Off Index értékkel. Legalább egy kontrollminta használata kötelező, a hiányzó Cut-Off Index értékek nullák lesznek.

- Lehetséges hibák a kiértékelési folyamat során:

  - A készülék nem azonosítható a megadott karakter sorozattal a mintaillesztési fázisban.

– A kívánatostól eltérő számú klaszter van jelen.
– A kvantitatív standard-ekhez nincs illeszthető görbe a megadott függvénnyel.

Amennyiben a vizsgálatok mentesek ezektől a matematikai jellegű hibáktól, akkor a kísérletet ki lehet értékelni biológiailag is. Az eredmények táblázatos formában tekinthetőek meg. Az előre megadott üzenetek segítségével a tesztminták eredménye könnyen értelmezhető. Mindezek együttesen biztosítják, hogy a biológiai területen kutatók gyorsan és egyszerűen tervezhetik meg és értékelhetik ki a kísérleteiket egy kényelmes környezetben. Az eredményeket pedig olyan formában lehet kinyerni, ami könnyen olvasható és értelmezhető.

1.6 A projekt jelenlegi állapota

A két program egyesítésre került és kibővült számos új funkcióval. Jelenleg a Beckton Dickinson cégnél van használatban, a flow-citométer készülékeikhez mellékelik.

2. Keresztplatformos fejlesztés

Számos kritikus kérdés merült fel a szoftverek Macintosh platformra való portolásakor. Az én felelősségem volt, hogy a legjobb megoldásokat megtaláljuk. Úgy gondolom, hogy a teljes csapat a legjobb tudása szerint dolgozott, és a Macintoshra portolt programok annyira hasonlítanak a PC-s verziókra amennyire az csak lehetséges.

A választott eszközök kielégítően működtek, nagyon jó csapatmunkát tettek lehetővé, mindamellett automatizálták az archiválási folyamatot. A C++ nyelv könnyen portolható, a standard függvényei és osztályai pedig kényelmesen használhatók.

A munka során néhány burkoló osztályt kellett létrehoznunk (nyomtatási, beállítási, stb.) és néhány vezérlő elemet át kellett ültetnünk Macintoshra (tooltip, szerkeszthető táblázat, üzenet ablakok, stb.).

3. Bittérkép transzformációk

A kezdeti célom az volt, hogy $m \times n$-es bittérképeket transzformáljak konvex négyszögre illetve olyan különböző felületekre, melyek síkba kifejthetőek. Ehhez tanulmányoztam a szükséges matematikai hátteret (lásd [11], [12], [13], [14],

[15]). A kettősviszony invarianciáját kihasználva sikerült válós idejű algoritmusokat kifejlesztenem (lásd [16], [17], [18], [19], [20], [21]). A bittérképeket síkbeli rácsnak tekintettem, és ennek a rácsnak a pontjait transzformáltam az új pozicióba. A bittérkép texeleit a sarkaiban található rácspontokkal írtam le. A transzformált texelek együttesen a transzformált bittérképet szolgáltatják.

Később további módszerek kifejlesztésével a szabad határvonalú alakzatokra is lehetővé tettem bittérképek transzformációját valós időben bizonyos megszorítások mellett. A szakirodalomban sok olyan eljárást leírnak, amit jól lehet használni a részproblémáknál (lásd [22], [23], [24], [25]). A hossz- és keresztirányban szimmetrikus alakzatokra való transzformálás a tartomány fokozatos darabolásával oldható meg. Az alakzat közepe rögzül az első lépésben, majd belülről kifelé haladva teljesen lefedhető az alakzat a bittérképpel. A teljesen szabad határvonalú alakzatokat pedig a határvonal kicsinyítésével, illetve bilineáris súlyfüggvény alkalmazásával lehet kezelni. Gömbre való transzformáláshoz síksorok projektív koordinátáit használtam trigonometrikus függvények helyett, így az algoritmus egyszerű és lényegesen gyorsabb lett.

A neurális hálózatok segítségével további, kevesebb megszorítással rendelkező szabad határvonalú alakzatokat is textúrázni lehet. Az általam kidolgozott módszer sajnos nagyon lassú, és így már nem alkalmas valós idejű animációk létrehozására. A szabad határvonalú alakzatokat jól paraméterezhető B-spline görbékkel lehet például leírni. A B-spline görbék paraméterezésével kapcsolatos problémákat egy referált cikkben tanulmányoztam.

Vannak olyan felület-illesztéssel kapcsolatos eredmények, melyek segítségével téglalap alakú területet lehet egy körön keresztül két lépésben megfeleltetni egyenes szakaszokkal határolt konkáv alakzatnak. Ha megfelelően rövid szakaszokat választunk a határvonalhoz, akkor gyakorlatilag akármilyen alakzat megjeleníthető egy pixelekből felépülő képernyőn. A képlet kiszámítása meglehetősen időigényes és mivel én a valós idejű transzformációkat kutattam, így ez az eredmény kívül esik a vizsgálatom tartományán.

Úgy tűnik, hogy Moore törvénye nem érvényes többé (lásd [26]). Mindamellett úgy gondolom, hogy a gépek fejlődésével az ilyen típusú algoritmusok fogják meghatározni a bittérképek szabad határvonalú alakzatokra való transzformációjának jövőjét.

3.1 A jelölések és a használt eszközök

Az 3.1. ábra mutatja az eredeti bittérképet, amit számos különböző alakzatra illetve felületre fogok transzformálni. Jelöljük az $m \times n$-es bittérkép csúcsait rendre A, B, C és D betűkkel, a transzformált pontokat pedig vesszővel. Legyen O_1 az $\overline{AB}$ egyenes végtelen távoli pontja. Legyen O_2 pedig az $\overline{AD}$ egyenes

végtelen távoli pontja. Tekintsük az $m \times n$-es bittérképet egy $(m+1) \times (n+1)$-es rácsnak.

Figure 3.1: Az eredeti bittérkép (400×300)

A következő fejezetekben euklideszi síkot fogok kezelni, ami ki van terjesztve a végtelen távoli pontokkal. A sík véges részéből indulok ki, azt képezem le a sík véges részére. A közbülső számítások során használom majd a projektív geometria eszközeit. Az euklideszi sík egy pontját $P(x, y)$-al tudjuk jelölni. Bevezethetünk P-re egy új jelölést a következő módon: $P(\frac{x_1}{x_3}, \frac{x_2}{x_3})$, vagy $P[x_1, x_2, x_3]$. Ily módon a végtelen távoli pontok kezelhetővé válnak. Ha $x_3 = 0$, akkor a három koordináta egy végtelen távoli pontot ír le. Ebben az esetben (x_1, x_2) vektor a végtelen távoli pont irányát mutatja.

3.2 Transzformációk projektív invariánsok alapján

Ebben a fejezetben a kettősviszony invarianciáját kihasználó projektív leképezéseket fogok leírni.

3.2.1 Konvex négyszögre való transzformáció

A transzformáció után a négy lehetséges eset egyikét kapjuk. Az eredeti bittérkép minden egyes $K_{i,j}$ texeléhez egy négyszöget kell rendelnünk. A 3.2. ábra az eredeti bittérképet mutatja egy konvex négyszögre transzformálva.

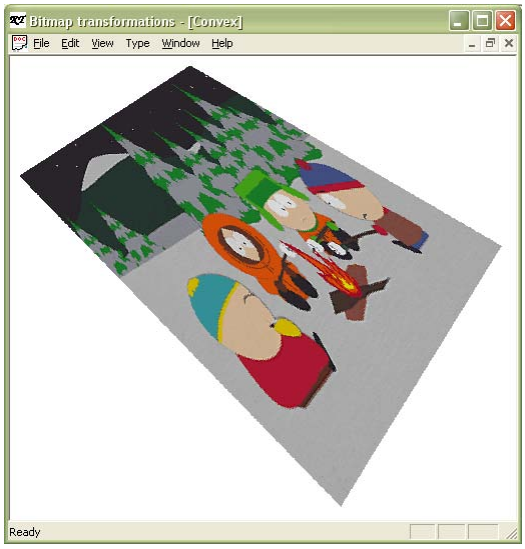


Figure 3.2: Az eredeti bittérkép konvex négyszögre transzformálva

Jelöljük az egyik határszakasz egy rácspontját P-vel és a transzformált rácspontot pedig P'-vel. Mivel a határszakaszokat egymástól függetlenül oszthatjuk fel n illetve m részre, ezért csak két esetet kell megvizsgálnunk:

1. A szemközti oldalak párhuzamosak egymással.
   Legyen P egy rácspontja az $\overline{AB}$ szakasznak. Ebben az esetben az $\overline{A'B'}$ és a $\overline{D'C'}$ szakaszokat az alábbiak alapján oszthatjuk fel n darabra:

$$\begin{aligned} (ABPO_1) &= (A'B'P'O_1') \\ (ABP) &= (A'B'P') \\ \frac{\overline{PA}}{\overline{PB}} &= \frac{\overline{P'A'}}{\overline{P'B'}} \end{aligned}$$

   Úgy kapjuk ezt az eredményt, hogy O_1 és O_1' végtelen távoli pontok.

2. A szemközti oldalak nem párhuzamosak egymással.
   Legyen P egy rácspontja az $\overline{AB}$ szakasznak. A P' helyét a kettősviszony invarianciája alapján tudjuk meghatározni. Az $(ABPO_1) = (ABP) = \frac{\overline{PA}}{\overline{PB}} = (A'B'P'O_1') = \frac{\overline{P'A'}}{\overline{P'B'}} : \frac{\overline{O_1'A'}}{\overline{O_1'B'}}$ és $\overline{P'B'} = \overline{A'B'} - \overline{P'A'}$ egyenletekből a következőket kapjuk: $\frac{\overline{PA}}{\overline{PB}} \cdot \frac{\overline{O_1'A'}}{\overline{O_1'B'}} = \frac{\overline{P'A'}}{\overline{A'B'} - \overline{P'A'}}$. Az utolsó eredményből a

$\overline{P'A'} = \frac{\overline{PA}}{\overline{PB}} \cdot \frac{\overline{O_1'A'}}{\overline{O_1'B'}} \cdot (\overline{A'B'} - \overline{P'A'})$ egyenletet kapjuk, amiből $\overline{P'A'}$ szakaszt kell kifejeznünk. $\overline{P'A'} \cdot (1 + \frac{\overline{PA}}{\overline{PB}} \cdot \frac{\overline{O_1'A'}}{\overline{O_1'B'}}) = \overline{A'B'} \cdot \frac{\overline{PA}}{\overline{PB}} \cdot \frac{\overline{O_1'A'}}{\overline{O_1'B'}}$, így $\overline{P'A'} = \frac{\overline{A'B'} \cdot \frac{\overline{PA}}{\overline{PB}} \cdot \frac{\overline{O_1'A'}}{\overline{O_1'B'}}}{(1+\frac{\overline{PA}}{\overline{PB}} \cdot \frac{\overline{O_1'A'}}{\overline{O_1'B'}})}$.

Ily módon az $\overline{A'B'}$ szakasz minden egyes P pontjának meg tudjuk határozni a P' képét. Ugyanilyen módon megkaphatjuk a $\overline{DC}$ szakasz képének pontjait.

Ha alkalmazzuk a metsző függvényt, egyenes sorok egyenleteit kapjuk. Ha metszük ezeket az egyeneseket azokkal, amelyeket az $\overline{A'D'}$ és $\overline{B'C'}$ szakaszokból nyertünk az előzőleg ismertetett módon, akkor az $m \times n$-es rács képét kapjuk.

3.2.2 Kifejthető felületekre való transzformálás

Csak olyan felületeket tudunk kezelni, melyeknek rendelkezésre áll a bijektív kifejtési függvénye. Először is a transzformációt meghatározó négy pontot kell a síkba transzformálnunk. A síkban meghatározzuk az eredeti rácspontok képét az előző fejezetben ismertetett módon. Ezután a síkbeli rács pontjait visszatranszformáljuk a kifejthető felületre. Így a texelekhez tartozó négyszögek összessége az eredeti bittérkép transzformáltját nyújtják a kifejthető felületre, ami lehet például forgáskúp vagy forgáshenger palástja (lásd 3.3. ábra).

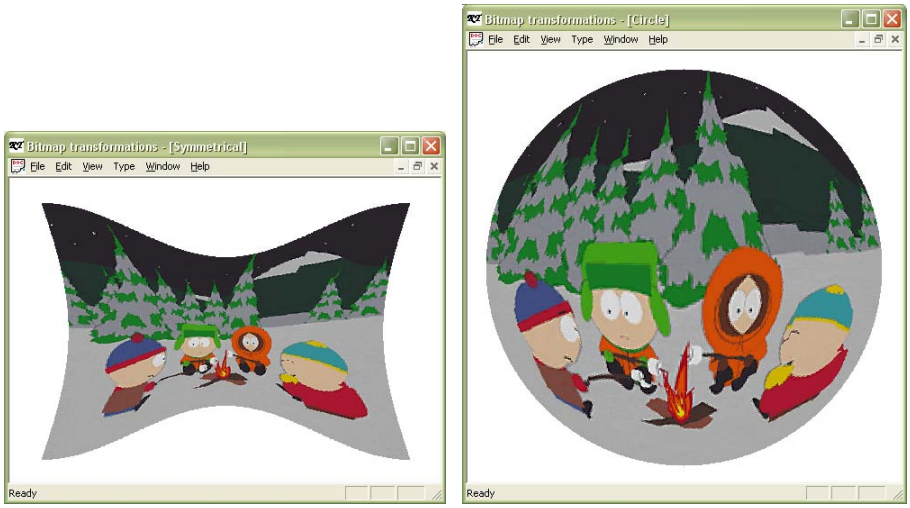
(a) Kúppalást (b) Hengerpalást

Figure 3.3: Transzformáció kifejthető felületekre

3.2.3 Projektív transzformáció merőleges szimmetriatengely párral rendelkező poligonra

A transzformációt a bittérkép folyamatos darabolásával tudjuk elérni. A kisebb partíciókat a 3.2.1. fejezet eredményei alapján tudjuk kitölteni. A particionálást úgy kell elvégezni, hogy a közös oldallal rendelkező partíciókra a következő teljesüljön: a közös oldalegyenes és a vele szemben elhelyezkedő oldalegyenesek egy közös metszésponttal rendelkezzenek. Az első egy vagy két lépés után a lehetőségeink elfogynak egy teljesen szabad határvonalú alakzat esetén. A 3.3. fejezetben ilyen esetekre mutatok be transzformációkat.

Az olyan alakzatok, melyek merőleges szimmetriatengely párral rendelkeznek tetszőleges sok lépésben particionálhatóak. Első lépésben az alakzatot négy részre osztjuk a szimmetriatengelyekkel, és a tengelyek metszéspontját rögzítjük. Ekkor a negyedeket tovább oszthatjuk négyfelé. A belső negyed rögzül, a többi részt továbboszthatjuk. Így az alakzatot belülről kifelé egyre kisebb négyszögekkel tudjuk lefedni, végeredményként a szimmetrikus alakzatot ki tudjuk textúrázni (lásd 3.4. (a) ábra), ami lehet például egy körlap vagy ellipszis (lásd 3.4. (b)).



(a) Általános szimmetrikus poligon (b) Körlap

Figure 3.4: Szimmetrikus poligonra való transzformálás

3.2.4 Transzformálás gömbre

A transzformált bittérképet egy félgömbön elhelyezkedő négy pont fogja meghatározni. Jelöljük a transzformált pontokat A', B', C', D', O'_1 és O'_2 betűkkel. A gömb sugarát r jelöli. Legyen $\underline{a'}$ az a vektor, ami O-ból - a kör és a koordinátarendszer középpontjából - A'-be mutat. Ha alkalmazzuk ezt a jelölést B', C' és D' pontokra is, akkor O'_1 és O'_2 pontokat a következőképpen kapjuk:

$$\underline{o'_1} = (\underline{a'} \times \underline{b'}) \times (\underline{d'} \times \underline{c'}) \qquad \underline{o'_2} = (\underline{a'} \times \underline{d'}) \times (\underline{b'} \times \underline{c'})$$

$$\underline{OO'_1} = r \cdot \frac{\underline{o'_1}}{|\underline{o'_1}|} \qquad \underline{OO'_2} = r \cdot \frac{\underline{o'_2}}{|\underline{o'_2}|}$$

A gömbi négyszöget gömbi főkör darabok határolják. Tekintsük a gömb azon metszetét, amit úgy kapunk, hogy az merőleges legyen $A'D'O$ és $B'C'O$ síkok metszetére és tartalmazza az O pontot.

Két síksorra van szükségünk: egyre az $A'D'O$ és $B'C'O$ síkok és egyre az $A'B'O$ és $D'C'O$ síkok között. A transzformált rácspontokat a gömb és a két síksor metszeteként kapjuk. Ez két elsőrendű projektív forma, egy síksor és egy pontsor közötti projektív leképezést jelent.

Egy új projektív koordinátarendszer alapsíkjai legyenek az $A'D'O$ és $B'C'O$ síkok. Az $A'D'O$ sík projektív koordinátái $(0, 1)$, a $B'C'O$ sík projektív koordinátái pedig $(1, 0)$ az új koordinátarendszerben. Az alapsíkok közös egyenese lesz a tartója egy síksornak. Fejezzük ki az $O'_1O'_2O$ síkot, ami (λ, μ) koordinátákkal rendelkezik az alapsíkok segítségével. μ legyen 2. Akkor is a megfelelő λ értéket kapjuk, ha síkok normálvektorát alkalmazzuk.

$$\begin{aligned} \underline{n_{O'_1O'_2O}} &= \underline{o'_1} \times \underline{o'_2} \\ \underline{n_{A'D'O}} &= \underline{a'} \times \underline{d'} \\ \underline{n_{B'C'O}} &= \underline{b'} \times \underline{c'} \end{aligned}$$

Mivel $\underline{n_{A'D'O}}$ és $\underline{n_{B'C'O}}$ legalább egy koordináta értékben különböznek, és μ egyenlő 2-vel, a következő három egyenlet legalább két független tartalmaz. Két ismeretlenünk van: k és λ.

$$\begin{aligned} k \cdot (\underline{n_{O'_1O'_2O}})_{(x)} &= \lambda \cdot (\underline{n_{A'D'O}})_{(x)} + \mu \cdot (\underline{n_{B'C'O}})_{(x)} \\ k \cdot (\underline{n_{O'_1O'_2O}})_{(y)} &= \lambda \cdot (\underline{n_{A'D'O}})_{(y)} + \mu \cdot (\underline{n_{B'C'O}})_{(y)} \\ k \cdot (\underline{n_{O'_1O'_2O}})_{(z)} &= \lambda \cdot (\underline{n_{A'D'O}})_{(z)} + \mu \cdot (\underline{n_{B'C'O}})_{(z)} \end{aligned}$$

Ezekből az egyenletekből k és λ könnyedén meghatározható. Minket csak a λ értéke érdekel.

Az $A'B'O$ és $D'C'O$ síkok által meghatározott síksor egy eleme $(x1_i, y1_i)$. Az $(x1_i, y1_i)$ sorozatot úgy kapjuk meg, ha a $\overline{AB}$ szakasz minden P_i rácspontját transzformáljuk. Így $(ABP_iO_1) = (ABP_i) = \frac{\overline{P_iA}}{\overline{P_iB}} = \frac{\lambda-1}{\lambda-0} : \frac{x1_i-1}{x1_i-0}$. Az előbbi eredményből $\frac{\overline{P_iA}}{\overline{P_iB}} \cdot \frac{x1_i-1}{x1_i} = \frac{\lambda-1}{\lambda}$ következik. $x1_i$-t kell kifejeznünk az $\frac{x1_i-1}{x1_i} = \frac{\lambda-1}{\lambda} \cdot \frac{\overline{P_iB}}{\overline{P_iA}}$ egyenletből. Mivel $x1_i \cdot (1 - \frac{\lambda-1}{\lambda} \cdot \frac{\overline{P_iB}}{\overline{P_iA}}) = 1$, így $x1_i = \frac{1}{1-\frac{\lambda-1}{\lambda} \cdot \frac{\overline{P_iB}}{\overline{P_iA}}}$. Hasonlóképpen $\frac{\overline{P_iA}}{\overline{P_iB}} = \frac{\mu-0}{\mu-1} : \frac{y1_i-0}{y1_i-1}$ és $\mu = 2$, így $\frac{\overline{P_iA}}{\overline{P_iB}} \cdot \frac{y1_i}{y1_i-1} = 2$. Az előző egyenlőségekből $y1_i = 2 \cdot (y1_i - 1) \cdot \frac{\overline{P_iB}}{\overline{P_iA}} = 2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}} \cdot y1_i - 2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}} = \frac{2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}}}{2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}} - 1}$ következik.

Ha tekintjük az $A'D'O$ és $B'C'O$ alapsíkokat, akkor a közös egyenesük szintén egy síksor tartója lesz. Legyen ezen síksor egy eleme $(x2_j, y2_j)$. Ekkor a P'_{ij} pont, amely az eredeti bittérkép P_{ij} rácspontjának a megfelelője, a gömb és az első síksor $(x1_i, y1_i)$ paraméterű és a második síksor $(x2_j, y2_j)$ paraméterű síkjainak a metszéspontja. Az algoritmusban érdemesebb a síkok normálvektorával számolni. Az eredményezett kép az 3.5. ábrán látható.

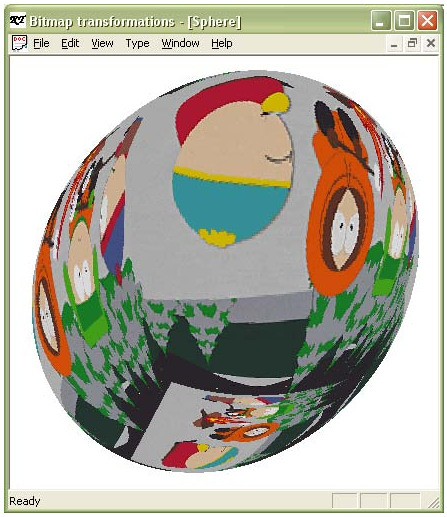


Figure 3.5: Hat bittérképpel lefedett gömb

Konklúzióként azt mondhatjuk, hogy gömbi négyszögre a kettősviszony ki-

használásával trigonometrikus függvények használata nélkül tudtam gyors algoritmust készíteni.

3.3 Módszerek szabad határvonalú alakzatokra

Ha nem csak szimmetrikus alakzatokat szeretnénk kezelni és folyamatos leképezést szeretnénk, akkor új módszereket kell kifejleszteni. A következő algoritmusok kiterjesztik a kezelhető alakzatok szabadságát.

3.3.1 A határvonal zsugorítása

A következőkben feltételezzük, hogy a bittérkép $N \times N$-es. A gyakorlatban ez nem megszorítás, mivel egy téglalap alakú bittérkép könnyedén $N \times N$-es alakra hozható. A transzformált határvonalakat a $(x_1(t_1),\ y_1(t_1))$, $(x_2(t_2),\ y_2(t_2))$, $(x_3(t_3),\ y_3(t_3))$, $(x_4(t_4),\ y_4(t_4))$ függvénypárok írják le, ahol $t_1,\ t_2,\ t_3,\ t_4 \in [0,1]$.

A négy csúcspont megfelelői a következők lesznek: $(x_1(1),\ y_1(1)) = (x_2(0),\ y_2(0))$, $(x_2(1),\ y_2(1)) = (x_3(1),\ y_3(1))$, $(x_1(0),\ y_1(0)) = (x_4(0),\ y_4(0))$ és $(x_4(1),\ y_4(1)) = (x_3(0),\ y_3(0))$. Legyen $f_i(t_i) = (x_i(t_i),\ y_i(t_i))$ és $a_i(t_i, s) = f_i(t_i) + (KP - f_i(t_i)) \cdot s/(N/2),\ i = 1,2,3,4$. Legyen KP az átlók metszéspontja. Az átlók az alakzatot négy részre osztják. A következő megszorításoknak kell teljesülnie minden $f_i(t_i)$ esetén: egyik $f_i(t_i)$ görbének sem lehet metszéspontja az átlókkal, emellett csak egy metszéspontja lehet a KP középpontból kiinduló egyenesekkel. Az $f_i(t_i)$ görbék megváltoztatása csak az adott negyedre van hatással.

Az eredeti rács transzformált határpontjai a következők:

$$\begin{array}{lllll}
(x_1(0), y_1(0)) & (x_1(1/N), y_1(1/N)) & (x_1(2/N), y_1(2/N)) & \ldots & (x_1(1), y_1(1)); \\
(x_2(0), y_2(0)) & (x_2(1/N), y_2(1/N)) & (x_2(2/N), y_2(2/N)) & \ldots & (x_2(1), y_2(1)); \\
(x_3(0), y_3(0)) & (x_3(1/N), y_3(1/N)) & (x_3(2/N), y_3(2/N)) & \ldots & (x_3(1), y_3(1)); \\
(x_4(0), y_4(0)) & (x_4(1/N), y_4(1/N)) & (x_4(2/N), y_4(2/N)) & \ldots & (x_4(1), y_4(1)).
\end{array}$$

A transzformált belső pontok az a_i függvény megfelelő paraméterezésével kaphatóak meg. Ezeket a pontokat a határvonaltól kiindulva a KP pont felé haladva határozhatjuk meg. Minden egyes lépésben kettővel kevesebb pontot kell meghatároznunk.

Az egyik negyedhez tartozó paraméterek a következők:

$$\begin{array}{lllll}
a_i(0,0) & a_i(1/N,0) & a_i(2/N,0) & \ldots & a_i(1,0); \\
a_i(0,1/(N/2)) & a_i(1/(N-2),1/(N/2)) & a_i(2/(N-2),1/(N/2)) & \ldots & a_i(1,1/(N/2)); \\
a_i(0,2/(N/2)) & a_i(1/(N-4),2/(N/2)) & a_i(2/(N-4),2/(N/2)) & \ldots & a_i(1,2/(N/2));
\end{array}$$

$$\ldots$$

$$a_i(0,(N/2-1)/(N/2)) \quad a_i(1/2,(N/2-1)/(N/2)) \quad a_i(1,(N/2-1)/(N/2));$$
$$a_i(1,1).$$

A negyedek közös pontjait elegendő egyszer kiszámolni. A módszer előnye a lokális változtathatóság. Bár a negyedek kitöltése független egymástól, a határvonal mentén illeszkedik a minta, de törés jöhet létre a mintában (lásd 3.6. ábra).

Figure 3.6: Szabad határvonalú alakzatra való leképezés

3.3.2 Bilineáris súlyfüggvény alkalmazása

A szabad határvonalú alakzatra való leképezést a következőképpen is értelmezhetjük: két függvénysorozatot kell definiálni két függvénypár között, és a metszéspontok szolgáltatják majd a transzformált rácsot (lásd [27]). Legyen az eredeti bittérkép $N \times M$-es.

Legyen $a_1(u)$, $a_2(u)$, $u \in [0,1]$, és $b_1(v)$, $b_2(v)$, $v \in [0,1]$ görbék adottak úgy, hogy egy síkban legyenek és az a görbéknek csak egy metszéspontja legyen a b görbékkel. Ezek a görbepárok határozzák meg a transzformált bittérképet. Célunk egy olyan $s(u,v)$, $u,v \in [0,1]$ függvény találása, amelyre $s(u,0) = a_1(u)$, $s(u,1) = a_2(u)$; $s(0,v) = b_1(v)$, $s(1,v) = b_2(v)$ igazak.

Függvénysorozatokkal oldhatjuk meg a problémát. Tekintsük az $a_1(u)$ és $a_2(u)$ által definiált $I_a(u,v) = (1-v)\cdot a_1(u) + v \cdot a_2(u)$ függvénysorozatot és a $b_1(v)$ és $b_2(v)$ által definiált $I_b(u,v) = (1-u)\cdot b_1(v) + u \cdot b_2(v)$ függvénysorozatot. Ezek a függvénysorozatok a szemközti görbéket interpolálják, azonban a végpontjaik nem követik a másik görbepárt. A négy csúcspont bilineáris interpolációja oldja meg ezt a problémát (lásd 3.7. ábra).

$$I_{ab}(u,v) = \begin{bmatrix} 1-u & u \end{bmatrix} \cdot \begin{bmatrix} s(0,0) & s(0,1) \\ s(1,0) & s(1,1) \end{bmatrix} \cdot \begin{bmatrix} 1-v \\ v \end{bmatrix}$$

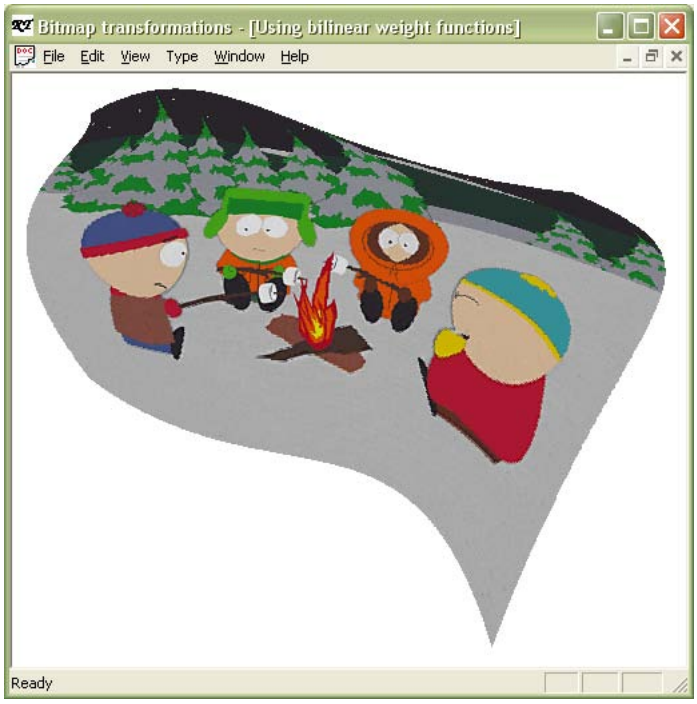


Figure 3.7: Transzformáció bilineáris súlyfüggvény segítségével

Az $s(u,v) = I_a(u,v) + I_b(u,v) - I_{ab}(u,v)$ függvény a megoldás. A transzformált rács a következő paraméterekkel áll elő:

$$\begin{bmatrix} s(0,0) & s(0,1/M) & s(0,2/M) & \dots & s(0,1) \\ s(1/N,0) & s(1/N,1/M) & s(1/N,2/M) & \dots & s(1/N,1) \\ & & & \dots & \\ s((N-1)/N,0) & s((N-1)/N,1/M) & s((N-1)/N,2/M) & \dots & s((N-1)/N,1) \\ s(1,0) & s(1,1/M) & s(1,2/M) & \dots & s(1,1) \end{bmatrix}.$$

Az $s(u,v)$ függvény azonban sajnos olyan pontokat is előállíthat, melyek kívül esnek az eredeti alakzaton. A határvonalak megadásánál ezt a tényt figyelembe kell venni.

A határvonalak módosítása az egész képre hatással van, azonban a részeredmények újrafelhasználásával az algoritmus felgyorsítható.

3.4 Neurális hálózatok alkalmazása

Ez a fejezet a neurális hálózatok alkalmazását mutatja be. Nagy előnye a módszernek, hogy tetszőleges topológia választható egy pixelhez (ami lehet hatszögletű vagy háromszög a szokásos négyzetes helyett), mivel a topológia nem változik az algoritmus használata közben. Nagy hátránya viszont, hogy az eddigi módszerekkel szemben nem alkalmas valós idejű transzformációk létrehozására.

3.4.1 Bevezetés a neurális hálózatokba

A neurális hálózatok elmélete egy nagyon hatékony eszköz (lásd [28], [29]). A következőkben egy Kohonen (lásd [30], [31], [32], [33]) hálózatot fogok használni az erős tanulási képessége miatt.

Az algoritmus két jól elkülönített részből áll majd.

1. A neurális hálózatot véletlen elhelyezkedésű pontokkal tanítjuk. A kezdeti rácspontok ily módon szétterjednek az alakzaton.

2. Az eredeti pixeleknek megfelelő területeket ki kell tölteni a megfelelő színnel.

3.4.2 A tanítási folyamat

A hálózat egyenletes eloszlású pontokkal lesz tanítva, lásd [30]. Ahhoz, hogy a megfelelő formát kapjuk, a határvonal pontjait is fel kell rendszeresen használni a tanítás során. Mivel a problémánk kétdimenziós, a bemeneti réteg két csúcspontból áll. A kimeneti réteg $m = (k + 1) \cdot (l + 1)$ elemből áll, ahol k és l az eredeti bittérkép sorainak és oszlopainak a száma. Ez az m kimeneti csúcspont egy rácsot alkot. A bemeneti réteg összes csúcspontja összeköttetésben van az összes kimeneti csúcsponttal. Az összeköttetéseknek súlya van: w_{ij} jelöli az i-ik bemeneti csúcspont és a j-ik kimeneti csúcspont közötti súlyt.

- A bemeneti pontok koordinátái a következők: $p_i(x_{1i}, x_{2i})$ $(i = 1, \ldots, n)$, ahol n a bemeneti vektorok száma.

- A kimeneti pontok koordinátái a következők: $q_j(w_{1j}, w_{2j})$ $(j = 1, \ldots, m)$, ahol m a hálózat csúcsainak a száma.

A tanítási folyamat a következőképpen zajlik:

1. lépés A w_{sj} súlyok inicializálása, $(s = 1, 2; j = 1, \ldots, m)$ a választott topológiának megfelelően. Legyen $t = 1$, ami a tanítási időt jelöli.

2. lépés Tekintsük az (x_{1i_0}, x_{2i_0}) új bemeneti értékeket, amelyek a tetszőlegesen választott p_{i_0} bemeneti pont koordinátái. Ügyelni kell a határvonal pontjainak rendszeres bevitelére.

3. lépés Kiszámítjuk az euklideszi távolságot kimeneti csúcspontok és a bemeneti pont között:

$$d_j = \sum_{s=1}^{2} (x_{si_0} - w_{sj})^2.$$

4. lépés Meg kell keresni a győztes q_{j_0} egységet, ami a minimális távolsággal rendelkezik a bemeneti ponthoz, ahol j_0 az érték, amire $d_{j_0} = min(d_j)$.

5. lépés Kiszámítjuk az $N(t) = (j_0, j_1, \ldots, j_k)$ szomszédságot.

6. lépés Frissítjük a szomszédság csúcsainak a súlyait a következő képlet alapján:

$$w_{sj}(t+1) = w_{sj}(t) + \eta(t) \cdot (x_{si_0} - w_{sj}(t))^2 \quad \forall j \in N(t),$$

ahol $\eta(t)$ a nyereség, ami idővel csökken.

7. lépés Legyen $t = t + 1$. Ismételjük meg a 2-7 lépéseket, amíg a hálózat nincs megtanítva. Ez azt jelenti, hogy a hálózat változása egy előre meghatározott határ alatt van.

A súlyok inicializálása

Úgy kell megadni őket, hogy a kezdeti rácspontok téglalapot alkossanak az alakzat közepén.

A sugár és a nyereség meghatározása

Mindkettőnek Gauss görbének kell lennie és időben csökkennie kell. Az algoritmusomban a következő képleteket használtam a sugár és a nyereség kiszámításához (lásd [32] és [33]):

$$Gain(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t}{q})^2} \quad Radius(t) = \frac{n}{2} e^{-\frac{1}{2}(\frac{t}{s})^2},$$

ahol t jelöli az iterációk számát, q és s léptékek az időtengelyen, míg n jelöli a bemeneti pontok számát.

A neurális hálózatok használatával szép képeket kaphatunk, azonban a sebesség nem kielégítő. Megfelelő paraméterezés esetén a pixelek képei nagyjából egyforma méretűek. A módszer használatával sikerült kibővíteni a textúrázható alakzatok körét.

1. MMA Technology

The MMA technology is an analytical method. It is used in for medical and experimental purposes. The aim of our team was to build up a complex method that is supported by automated computer programs. This chapter gives the problem's description and skeleton of the solution how to handle Multiplex Microbead Assay measurements.

1.1 Principles of the MMA Technology

The advantage of the MMA technology is that several reactives can be used in a measurement instead of one, provided that several stains are used with homogeneously mixed beads that are covered with various reactives (see [1], [2], [3]). This way the necessary amount of sample (blood, plasma or tincture) will be significantly smaller. The beads are identified by the emission of the contained fluorescent stain. If only one stain is used with numerous concentrations, then maximum 10-16 beads can be distinguished (with little overlay) from the data of one channel. Two channels' data is needed when two stains are used, so the maximum number of distinguishable beads rise to 256. The number of distinguishable beads can be extended by using different sized beads. The software firstly recognize the same sized beads, which is done with the help of the side scatter channel (SSc). The second step is to distinguish the events by analytes using one or two dimensional area division method. The segregated patches are called clusters.

1.1.1 Aim of the Developed Softwares

The aim of the developed softwares is to help doctors and researchers in using multiple beads during measurements (see [4], [5], [6], [7], [8]). Although the beads were ready for a while for use, there were no fashioned way for using them. Doctors and researchers have made up better or worse own systems, and applied them by many paperwork and computations by hand. We wanted to give a well-fashioned schema and support it by automating the process by the developed programs as far as it is possible.

ASF Creator supports the creation of an Assay Specification File that comes with a kit. In turn, the SAT Flow software supports a whole measurement process, where one or more kit can be used for the samples.

The software supports flow-cytometers which are mainly used by researchers

for various measurements. The current software version of SAT Flow does not have license to use it in medical treatments.

1.2 Steps of a Measurement

The guiding principle is as follows:

- First of all, the assay to be examined, which can be a group of cells or a microbead mixture, mixed with the appropriate reagents.
- After this step, the obtained mixture is mixed with an organic fluorescent stain that is connecting to the reagent.
- The mixture inserted into the machine.
- The diluted mixture goes into the cuvet (exposition box) through a capillary.
- The high speed mixture is lit by laser in this box. The laser light is reflecting in different directions or absorbing in it. In the second case the laser excites the stain molecules, so they begin to emit light.
- The 'event' phrase will be used for the assay that has just passed by the laser beam.
- The intensity of the reflected light is perceived by some well placed sensors.
- The signs are forwarded to the cytometer's processing unit. The signs are digitized here. Older machines use 8, 10, 15, 16 bits, the newest models are capable to use 32 bits. Usually a logarithmic conversion is possible to achieve the best results.
- The digitized channel values get to the collector machine. A List Mode file (abbreviated form: LM file) is created for each measurement.

1.3 The ASF Creator Software

The new assay kit can defined by giving all data or by modifying an existing kit's data. A new assay kit can support one of the qualitative or quantitative models.

The producer of the kit assigns a name and an ID to the assay that will be the identifier of it. LOT is a unique number that identifies a batch. The Name - ID - LOT triple uniquely identifies a kit. Assay type defines the evaluating

mode. The number of clusters determines how many clusters will be in the AS file. The clustering parameters will define the clusters. Expiration date is the creation date and the default expiration interval.

Only those instruments' LM files will be recognized by the SAT Flow software that are listed here with the proper parameter order. Instruments can be defined by LM files made with the instrument to be added.

The next step is to specify the file holding the data of the assay kit. The LM (List Mode) file has to be created with one of the known instruments. The parameters take the default value from the instrument list.

When the automatic processing fails that means there are not enough clusters, a message box warns the user to review the clustering parameters. If the clustering is done without problem, the user has to pair the analytes and the beads.

1.3.1 Quantitative model

Until this step, the wizard pages of the Quantitative and Qualitative models are the same.

In Quantitative model each analyte has to be assigned to a fitting equation. For example cubic, four parameter logistic and exponential sigmoid functions can be selected.

Standards will yield the base points of a curve that converts the intensity values into concentrations. The fitting functions of the quantitative model determine their parameters based on the measured intensity values and the given concentrations of the standards. They fit a curve on the yielded points. Putting in the test samples intensity values in this function we get the corresponding concentrations. A kit can contain arbitrary number of standards, but since the standard samples are very expansive, a reasonable number of them is expected.

The quantitative control samples can be used for verification purposes. They have three concentrations: low, medium and high. The Low C, Medium C and High C columns store the nominal concentration values for the analytes. The Low dx, Medium dx and High dx columns store the allowed deviance from the nominal concentration values in percentages. The message column has the warning texts that appears in the reports of the SAT Flow program when the concentrations are outside of the proper interval. The Number of replicates fields gives the repetition number of the control samples.

When the concentration of an analyte satisfies certain conditions, then the user is prompted a message. This can be text of the diagnosis for example. These messages can be defined on the next wizard page.

On the last option page of the wizard it can be set how many instrument standard beads will be in the kit. Their Relative Fluorescent Intensity (RFI) can

be given separately. These standards can have arbitrary number of replicates, only the cost limits this number.

The last information page summarizes the choices made so far. The user can print and save the kit to file here. Furthermore, there is a possibility to set up the printer or get a preview of the printed kit documentation. The print preview shows how the official printout will look like.

When the kit is saved, the Save... button is greyed. A kit can be saved only once. Then the kit can be printed or the user can decide to completes the kit by pressing the finish button. If the kit is not saved yet, the user is prompted a warning message that there is no original printout of the kit. Printing is not obligatory, but strongly recommended.

1.3.2 Qualitative model

In qualitative model the next page is the data specification for the model. The user can define the positive and negative control samples separately. It is common that only one of them is defined in a kit. The replicate number of them is given respectively.

The next two steps are identical to quantitative case. The user has to give the reporting messages for the analytes and define the instrument standard beads similar to the quantitative branch. The most important difference is that these messages are based on Cut-Off Indices instead of concentration values. The final saving and printing process are the same, too.

1.4 AS Files

The ASF Creator software yields an AS file that supports a kit. It contains:

- Type of the evaluation (qualitative, quantitative, IS)
- A name that describes the function
- Producer defined catalog ID
- Series identifier (LOT number)
- Unique identifier number
- Names of the supported cytometers
- Character series that identifies the cytometer in a List Mode file's header
- Stain (clustering) identifier parameters

- Size (SSc) identifier parameter
- Analytical (reporter) parameter
- Reactive (analyte) list in the order defined by the clustering procedure
- Bead list in the order defined by the clustering procedure
- Conditional messages' types, bounds and the texts of the messages

In quantitative case:

- The type of the function to be fitted
- Concentrations of the standard samples by analytes and the unit of the concentration

In qualitative case:

- Cut-Off Indices (COI) of the positive and negative control samples by analytes and the tolerance

In Instrument Standard case:

- Lack of the clustering parameters, stain list, control samples and conditional messages
- Relative Fluorescent Intensity (RFI) value for each bead

1.5 The SAT Flow Software

Steps of an experiment:

- The researcher decides the type of the experiment. He specifies the reactives and decides whether qualitative or quantitative measurement is needed. Instrument standards can be used to verify the integrity of the cytometer.
- The researcher selects the kit that satisfies his needs and imports the AS file of the kit into the SAT program.
- Assays of a kit can be used in different ways.
  - Batch assay: A given test has to be applied for samples obtained from many various sources. The end user can place test samples in a Batch assay.

  - Random assay: Used for patients. Assays for the different tests have to be assigned to the appropriate patients. Then, many different tests can be applied to a given sample obtained from one source.

- A researcher can add and modify test samples to an assay. The amount of the other samples is defined in the AS file. These samples that can not be modified are:

  - Standards - base points of the curve used for convert the concentration of intensity at quantitative evaluation. An intensity value associated with this sample in the LM file having known concentration gives the second coordinate of the base point.
  - Quantitative Control - used for verification, shall be in a range, else the evaluation process gives an error message.
  - Qualitative Control - the qualitative controls have similar role to the standards of the quantitative model. They determine the parameters of a function that is necessary to compute the Cut-Off Indices.
  - Instrument Standard - unstained calibrating beads. The usage of the instrument standards can be turned off.

- The samples can be arranged on a Microbead Array Plate, or on a Rack. A plate has 96 or 384 wells in a matrix arrangement. For better results, replicates can be asked for any sample, where the replicate is from the same tincture. Then, a map can be printed for the technician. This map shows which tincture must inserted in which well.

- The technicians accomplish the measurements.

- LM files shall be copied into one folder and the Data File Assignment option can be used for associating assay replicates with List Mode files. Automatic assignment can be used. This assignment can be modified manually if necessary.

- The report's form and appearance farthest at this point must be decided. Two kind of report can be generated, a shorter 'Summary Report' and a detailed 'Full Report'.

- After everything is set up, the evaluating procedure can be started, so the List Mode files get processed. Clustering is done due to the device parameters fixed in the AS file, this means the selection of the events of one reactive. The qualitative and quantitative methods use different evaluating procedure. In quantitative case, there is a curve fitting for

the standards with the curve fixed in the AS file. With the help of the obtained curve parameters the software defines the concentration values of all other samples and classifies the control samples. In qualitative case, a cut-off index is computed from the positive and negative control samples' intensity. The test samples' intensity values get divided with this cut-off index. One control sample is mandatory, and the missing cut-off indices will be zero values.

- Possible errors during the evaluation process:
  - The instrument cannot be identified with the help of the given character string at the pattern matching phase.
  - There is different number of clusters than should be.
  - There is no proper curve to fit with the given function for the intensity values of the clusters at quantitative standards.

  If the assays are free of these mathematical errors then the experiment can be evaluated biologically. The results are represented in a tabular form. The most important results are the derived concentrations, the derived cut-off indices and the deviation from the expected values at the control samples. With the help of the reporting messages the results of the test samples can easily be interpreted. All these tools together yield that a biologist researcher can plan and evaluate his experiment quickly and easily in a comfortable environment and gets the results in a form that can be read and interpreted easily.

1.6 Current State of the Project

The two programs were united and enhanced by several new features. It is in use at Beckton Dickinson, they enclose it to their flow-cytometer machines.

2. Cross Platform Developing

Since there are a lot of Apple Macintosh machines in the laboratories from the time when these kind of computers were the only choice, we had to support this platform, too. Nowadays IBM PCs are getting widespread even in this field, so the PC became the primary target.

We developed the PC version in advance and when a concept survived by time, then we have adopted the feature in the Mac version. There were tasks that could made parallel with the software planning and the early development

stages. For example: getting to know the new platform and its philosophy, possibilities and disadvantages. In general, one can say that creating the executables for a new platform takes one man's work in a four men project.

2.1 Choosing the Programming Language and the Developing Tools

This step is one of the most important task. The selected developing tools determine the whole process of implementing the software plans. Many aspects arise at choosing the right programming language. This is particularly true when the software has to support many platforms. When the language was fixed, we had to solve the task of the version control and the archiving of the source codes and other resources. The strongly object oriented Java has not satisfied our security and performance needs. Finally, C++ was the natural choice for the programming language, because the shared classes needed only a few modifications to get compiled on both platforms. Furthermore, we could very conveniently reuse a lot of codes written in C inherited from a former project, where LM files were processed.

After inspecting the existing developing tools, we have decided to use Microsoft's Visual Studio 6.0 on PC which has several advantage: program development has the support of MFC (Microsoft Foundation Classes) standard. The programs generated with the help of the MFC is not only programmed quicker, but furthermore they have the standard look of the Windows programs as well.

For the Macintosh platform the Codewarrior integrated development environment (IDE) seemed to be the best solution. It has a class hierarchy that is similar to the MFC and has a complex resource handler. As Visual Studio yielded the standard look of a Windows program, Codewarrior yields the standard look of a Macintosh program. We had to create an interface for the old graphical system and a little modified one for the new visual interface, called Aqua, that was totally rewritten in OS X.

Version control and archiving had to be solved in a proper way. We have used the Source Safe subsystem of the Visual Studio to keep track of the consecutively changing codes. Besides, this system can archive the project.

2.2 Differences Between the Platforms

A Macintosh program follows strict standards that are laid down in the Apple's Human Interface Guide (HIG) (see [9]). In the present days this strict rules

are getting less importance at software design while Windows programs always adapted the new techniques more easily.

The two platforms have slightly different messaging mechanisms. The messages are collected in message queues. The messages are handled by the appropriate functions of the system and the application.

The platforms have differences between the window systems' elements. An essential part of a Macintosh system is a mouse having only one button from the beginning. This led us to use only the left button of a PC's mouse, too.

2.3 Security Issues

Our programs are used for medical and research purposes, so security issues play a great role at the planning and programming phase. The files saved to disks are encrypted, and only one original printed copy can exists of the AS or SAT files that are time-stamped and signed manually by the responsible person.

Before a user could do anything, he shall authorize himself. There is an administrator user having a hardwired password, this user can create new users or delete old ones. The administrator user can not be deleted.

Saving and loading is standardized and work in the same way on different platforms. The strings are saved encoded by UNICODE for supporting more languages, and the numerical types follow the PC standards. The files are encrypted in order to protect the stored information.

At printing one topic is very important: only one original printed copy shall exist signed by the responsible person. All other copies of the original are marked with caption 'Copy' on each page.

2.4 Interface Planning

During the planning phase we have kept in mind the standards and the most common solutions. It became clear that it is advisable to set up a linear order to make sure that the user noticed all changes related to the last action. With careful design, we managed to order the option pages in such a sequence, that a setting does not affect the previous pages' settings. Our programs try to help the user by providing him with as many information as possible. The independent features can be set up from the menu directly too.

Windows systems have three basically different type windows: modal window, modeless window and float window. Macintosh systems have only one kind of window, but this window has many features. By setting these features in a proper way, all window types can be emulated.

In ASF Creator, the File and Edit menus have the standard elements. The toolbar and the icons are used only on PC.

2.5 Localization

Localization means not only the change of the texts that appear on the screen and in printing, but it involves the order of a date's elements, the separators, the decimal point, etc.

There was a need to support more than one language, especially the Japanese. This means that the ASCII7 is not enough, a multi byte character set is needed. UNICODE seemed to be the solution.

MFC specifically supports double-byte character sets (DBCS) from Multi-Byte Character Sets.

Unicode is a worldwide character-encoding standard that uses 16-bit character values to represent all the characters used in modern computing, including technical symbols and special characters used in publishing.

Fortunately, there are standard ANSI C functions that format a date in the proper order of the elements, yield the name of the months in the local language, etc.

2.6 Porting Problems and Solutions

Porting problems cover many questions that arise during the migration between the different platforms. From the byte order and representation standard of the float numbers through string handling to the classes of the development tools many difficulties has to be answered.

IBM PCs and Macintosh computers have different byte orders for floating point data types. With a simple macro this problem can be handled efficiently.

Since even in nowadays the written text is the base of the remote human communication, strings are extensively used in computer programs. ANSI C functions helps the tasks related with strings. Visual Studio has a class named CString to collect these functions together, while Codewarrior serves the same purpose with LString.

Message boxes are the basic form of the communication with user in decision-making situations. There are standard message boxes with different number of buttons in MFC on PCs. I have implemented an adaptive message box class for Macintosh that resizes to the included text and has the proper buttons.

On Windows, programs frequently use tooltips to explain buttons, areas or to display hidden information in boxes where scrollbar would have been used.

On the contrary, Macintosh does not even has such a control. We have found it handy and by implementing a tooltip class for the Macintosh we extended the possibilities.

The printing strategy is very different on the two platforms. We have created a class that supports starting and ending a printing process. The class implements many methods from the primitives to print out texts or graphics to the complex functions as print out heading, footer or complex boxes arranged in an optimal manner. In a word one can say that the printing functions are part of a wrapper class.

The programs usually have to store preferences that makes work easier. Windows uses a central database called Registry. On the other hand, Macintosh has a folder named Preferences where programs create a file with their names and store their preferences values in these files. Similar to the printing system, we have created a wrapping class that manages the Registry entries on PC and writes to or reads from the Preferences file on Macintosh.

The core or common classes caused only a few problem, because the C++ language platform independent and can be ported easily.

3. Bitmap Transformations

The aim of my research is to create algorithms which are able to transform bitmaps in real-time (see [11], [12], [13], [14], [15]). The algorithms developed so far are able to accomplish the transformation of an $m \times n$ bitmap to either a convex polygon or a spherical surface or a cone patch or a cylinder patch or a free-form patch. These are simple projective mapping to the surfaces (see [16], [17], [18], [19], [20], [21]). In the literature there are several procedures that can be applied at the minor problems (see [22], [23], [24], [25]). In the sphere case, the projective co-ordinates of plane sequences are used instead of using sinus functions, which are very comfortable and fast. For the free-form case I have developed various techniques. The free-forms can be defined by well-parametrized B-spline curves, which I have examined in a reviewed article.

3.1 Notation and the Used Tools

Figure 3.1 shows the original bitmap that will be transformed to numerous shapes and surfaces. Let us indicate the vertices of this $m \times n$ bitmap with A, B, C and D. Let O_1 be the point of infinity of the straight line $\overline{AB}$. Let O_2 be the point of infinity of the straight line $\overline{AD}$. Let us consider an $m \times n$ bitmap as an $(m+1) \times (n+1)$ grid.

Figure 3.1: The original bitmap (400×300)

In the following sections I will handle Euclidean planes that are extended with the points of infinity. I start from the finite part of the plane and I transform it to the finite part of the plane. Meanwhile the tools of the projective geometry will be used. A point of an Euclidean plane can be denoted by $P(x, y)$. We can introduce a new notation of P that will be the following: $P(\frac{x_1}{x_3}, \frac{x_2}{x_3})$, or $P[x_1, x_2, x_3]$. This way the points of infinity can be handled. When $x_3 = 0$, then the three co-ordinates describe a point of infinity. In this case (x_1, x_2) shows the direction of the point of infinity.

3.2 Transformations Using Projective Invariants

In this section I will describe projective mappings based on the invariance of the cross-ratio.

3.2.1 Transforming to a Convex Polygon

After the transformation of the bitmap to a convex polygon (see Figure 3.2), we obtain one of the four cases. We have to find a quadrilateral for each $K_{i,j}$ pixel of the original bitmap.

The four vertices are points of intersections of lines which are defined by four pairs of points. The first pair of these points fit on the $\overline{A'B'}$ section, the second

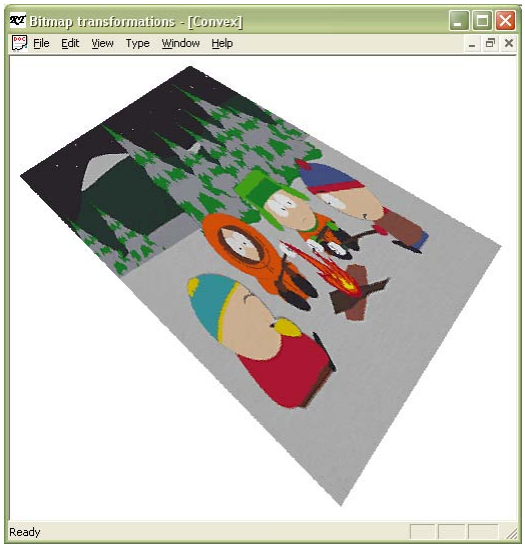


Figure 3.2: The original bitmap transformed to a convex polygon

pair of these points fit on the $\overline{D'C'}$ section, the third pair of these points fit on the $\overline{A'D'}$ section and the forth pair of points fit on the $\overline{B'C'}$ section.

Let us indicate the actual grid-point of a border section with P and the transformed grid-point with P'. Since we can divide into n pieces the sections independently, we have to examine only two cases:

1. Opposite sides are parallel to each other.
   Let P a grid-point of the $\overline{AB}$ section. In this case we have to divide the $\overline{A'B'}$ and the $\overline{D'C'}$ section into n pieces according to the following:

$$\begin{aligned} (ABPO_1) &= (A'B'P'O_1') \\ (ABP) &= (A'B'P') \\ \frac{\overline{PA}}{\overline{PB}} &= \frac{\overline{P'A'}}{\overline{P'B'}} \end{aligned}$$

   We get this result because O_1 and O_1' are points of infinity.

2. Opposite sides are not parallel to each other.
   Let P a grid-point of the $\overline{AB}$ section. The location of P' has to be defined according to the invariance of the cross-ratio. We get the following result from $(ABPO_1) = (ABP) = \frac{\overline{PA}}{\overline{PB}} = (A'B'P'O_1') = \frac{\overline{P'A'}}{\overline{P'B'}} : \frac{\overline{O_1'A'}}{\overline{O_1'B'}}$ and $\overline{P'B'} =$

$\overline{A'B'}-\overline{P'A'}$ equalities: $\frac{\overline{PA}}{\overline{PB}}\cdot\frac{\overline{O_1'A'}}{\overline{O_1'B'}}=\frac{\overline{P'A'}}{\overline{A'B'}-\overline{P'A'}}$. From the last result we get $\overline{P'A'}=\frac{\overline{PA}}{\overline{PB}}\cdot\frac{\overline{O_1'A'}}{\overline{O_1'B'}}\cdot(\overline{A'B'}-\overline{P'A'})$ equality from which we want to express $\overline{P'A'}$. $\overline{P'A'}\cdot(1+\frac{\overline{PA}}{\overline{PB}}\cdot\frac{\overline{O_1'A'}}{\overline{O_1'B'}})=\overline{A'B'}\cdot\frac{\overline{PA}}{\overline{PB}}\cdot\frac{\overline{O_1'A'}}{\overline{O_1'B'}}$, so $\overline{P'A'}=\frac{\overline{A'B'}\cdot\frac{\overline{PA}}{\overline{PB}}\cdot\frac{\overline{O_1'A'}}{\overline{O_1'B'}}}{(1+\frac{\overline{PA}}{\overline{PB}}\cdot\frac{\overline{O_1'A'}}{\overline{O_1'B'}})}$.

In this way we can define a P' point to each P point on the $\overline{A'B'}$ section. In the same way we can define the projected slide of the points of the $\overline{DC}$ section.

If we apply the intersecting function, we get equalities of a line sequence. Intersecting these equalities with those we defined from $\overline{A'D'}$ and $\overline{B'C'}$ the same way, we get the projected slides of the points of the $m \times n$ grid.

3.2.2 Transformation onto Developable Superficies

Only those superficies can be handled that have known bijective developing function. First, we have to transform the four points defining the location of the transformed bitmap to the plane, where we perform the definition of the points of the bitmap-grid with the help of the above-expounded method. After this step, we transform the points of the grid back to the surface. Thus the quadrilaterals give us the slide of the bitmap projected for example onto either cone of rotation (see Figure 3.3 (a)) or circular cylinder (see Figure 3.3 (b)).

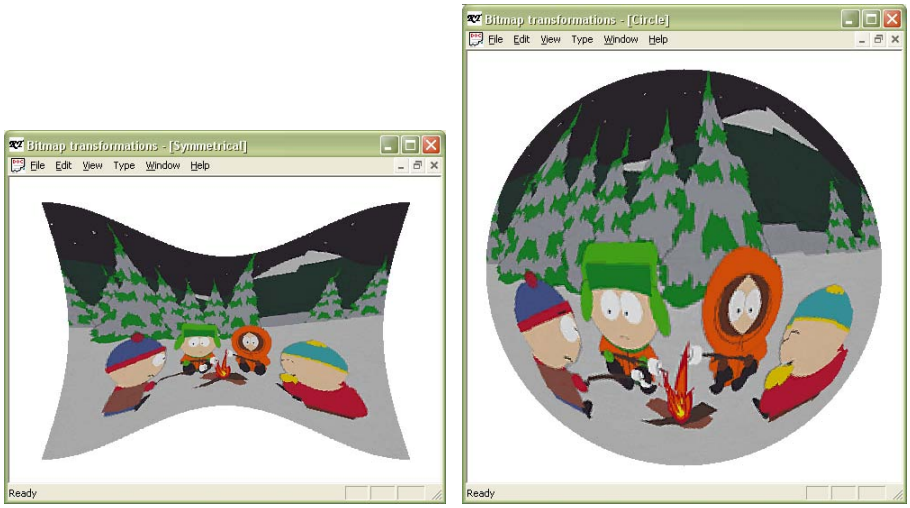
(a) Cone of rotation (b) Circular cylinder

Figure 3.3: Transformation onto surfaces that can be laid out

3.2.3 Projective Transformation to a Polygon Having Orthogonal Symmetric Axes Pair

We can divide the bitmap into smaller partitions. The projection to these smaller partitions can be done according to the result of Section 3.2.1. The transformed vertices of this partitions have to be defined so that for every two partitions having a common side, the following condition is true: the common side and the opposite sides to this common side must have the same point of intersection. After the first one or two steps we cannot modify anything. In Section 3.3 alternative methods will be introduced for transforming to free-form patches.

Special patches can easily be created that are symmetrical longwise and crosswise as well and satisfy the above condition. During the partition, we can change the position of the vertices of the larger partition. After dividing this partition into four parts equally, the previous vertices and the point of intersection of the diagonals will be fixed. The position of the other vertices can be changed. If we apply this method to the four smaller partitions of the original bitmap the same way but symmetrically, then we defined the projection of the original bitmap to a symmetrical patch (see Figure 3.4 (a)), for example to a circle (see Figure 3.4 (b)) or ellipse.



(a) General symmetric polygon (b) Circle

Figure 3.4: Transformation to symmetric polygons

3.2.4 Transforming to a Sphere

The situation of the transformed bitmap is defined by four points which have to be placed on the same half sphere. Let us indicate the transformed points with A', B', C', D', O'_1 and O'_2. r will be the radius of the sphere. Let $\underline{a'}$ be the vector that points to A' from O which is the center of the sphere and the co-ordinate system, too. If we apply this notation to B', C' and D' points as well, we can obtain O'_1 and O'_2 for given A', B', C' and D' the following way:

$$\underline{o'_1} = (\underline{a'} \times \underline{b'}) \times (\underline{d'} \times \underline{c'}) \qquad \underline{o'_2} = (\underline{a'} \times \underline{d'}) \times (\underline{b'} \times \underline{c'})$$

$$\underline{OO'_1} = r \cdot \frac{\underline{o'_1}}{|\underline{o'_1}|} \qquad \underline{OO'_2} = r \cdot \frac{\underline{o'_2}}{|\underline{o'_2}|}$$

The spherical quadrilateral, on which the projection is accomplished, is defined by those segments of the spherical main circle which is the intersection of the plane determined by the center of the sphere and the proper vertical pairs and the sphere.

Let us consider that intersection of the sphere which is received so that it is perpendicular to the line of intersection of the $A'D'O$ and $B'C'O$ planes and let the O point, which is the center of the sphere, fit into this intersection.

We need two plane-sequences: one between the $A'D'O$ and $B'C'O$ planes and one between the $A'B'O$ and $D'C'O$ planes. The transformed grid-points will be the intersection points of the sphere and the two plane-sequences. This means a projective mapping between two first degree projective basic forms: between a plane-sequence and a point-sequence.

We can calculate a plane that will be element of the first plane-sequence and contains P' point for every P grid-point of the $\overline{AB}$ section. The same way we can calculate a plane that will be element of the second plane-sequence and contains P' point for every P grid-point of the $\overline{AD}$ section.

Let the basic planes of a projective co-ordinate system be the $A'D'O$ and $B'C'O$ planes. $A'D'O$ will have $(0, 1)$ projective co-ordinate values and $B'C'O$ will have $(1, 0)$ projective co-ordinate values in the new co-ordinate system. The line of intersection of these basic planes will be the holder of the plane sequence with the help of which we can define the locations of the points transformed to the surface of the sphere. Let us express the $O'_1O'_2O$ plane having (λ, μ) co-ordinate values in the new co-ordinate system in terms of the basic planes. Let μ be 2. We will get the proper λ value as well, if we use the normal vectors

of the planes instead of the equalities of the planes.

$$\begin{aligned} \underline{n_{O'_1O'_2O}} &= \underline{o'_1} \times \underline{o'_2} \\ \underline{n_{A'D'O}} &= \underline{a'} \times \underline{d'} \\ \underline{n_{B'C'O}} &= \underline{b'} \times \underline{c'} \end{aligned}$$

Since $\underline{n_{A'D'O}}$ and $\underline{n_{B'C'O}}$ differ from each other at least in one co-ordinate value and μ is 2, the following three equations will contain at least two independent equations. There are two unknown values: k and λ.

$$\begin{aligned} k \cdot (\underline{n_{O'_1O'_2O}})_{(x)} &= \lambda \cdot (\underline{n_{A'D'O}})_{(x)} + \mu \cdot (\underline{n_{B'C'O}})_{(x)} \\ k \cdot (\underline{n_{O'_1O'_2O}})_{(y)} &= \lambda \cdot (\underline{n_{A'D'O}})_{(y)} + \mu \cdot (\underline{n_{B'C'O}})_{(y)} \\ k \cdot (\underline{n_{O'_1O'_2O}})_{(z)} &= \lambda \cdot (\underline{n_{A'D'O}})_{(z)} + \mu \cdot (\underline{n_{B'C'O}})_{(z)} \end{aligned}$$

From these equations k and λ can be determined easily. We are only interested in the value of λ.

Let an element of the plane sequence defined by $A'B'O$ and $D'C'O$ planes be indicated by $(x1_i, \underline{y1_i})$. We can produce this $(x1_i, y1_i)$ sequence if we take all P_i grid-point on the $\overline{AB}$ section, because the cross-ratio is invariant facing this type of projective projection. Thus $(ABP_iO_1) = (ABP_i) = \frac{\overline{P_iA}}{\overline{P_iB}} = \frac{\lambda-1}{\lambda-0} : \frac{x1_i-1}{x1_i-0}$. From this result $\frac{\overline{P_iA}}{\overline{P_iB}} \cdot \frac{x1_i-1}{x1_i} = \frac{\lambda-1}{\lambda}$ follows. We have to express $x1_i$ from $\frac{x1_i-1}{x1_i} = \frac{\lambda-1}{\lambda} \cdot \frac{\overline{P_iB}}{\overline{P_iA}}$. Since $x1_i \cdot (1 - \frac{\lambda-1}{\lambda} \cdot \frac{\overline{P_iB}}{\overline{P_iA}}) = 1$, so $x1_i = \frac{1}{1-\frac{\lambda-1}{\lambda} \cdot \frac{\overline{P_iB}}{\overline{P_iA}}}$. The same way $\frac{\overline{P_iA}}{\overline{P_iB}} = \frac{\mu-0}{\mu-1} : \frac{y1_i-0}{y1_i-1}$ and $\mu = 2$, so $\frac{\overline{P_iA}}{\overline{P_iB}} \cdot \frac{y1_i}{y1_i-1} = 2$. From the above equalities $y1_i = 2 \cdot (y1_i - 1) \cdot \frac{\overline{P_iB}}{\overline{P_iA}} = 2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}} \cdot y1_i - 2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}} = \frac{2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}}}{2 \cdot \frac{\overline{P_iB}}{\overline{P_iA}} - 1}$ follows.

If we consider $A'D'O$ and $B'C'O$ planes the basic planes, then the line of intersection of these planes will also be the holder of a plane-sequence. Let us indicate an element of this plane-sequence by $(x2_j, y2_j)$ parameters. So, the P'_{ij} points adequate to the P_{ij} grid-points of the original bitmap can be calculated as the point of intersection of the sphere and the line of intersection of the plane having $(x1_i, y1_i)$ parameters related to the first basic plane pair and the plane having $(x2_j, y2_j)$ parameters related to the second basic plane pair. During the producing of the algorithm, it is worth calculating with the normal vectors of the planes, because we can get the projected points of the grid points of the bitmap as the vectorial product of the normal vectors of the appropriate planes of the plane sequences having length r.

Consequently, we could transform to spherical surface taking advantage of the invariance of the cross-ratio without using trigonometrical functions in the algorithm, which are too time-consuming for computers (see Figure 3.5).

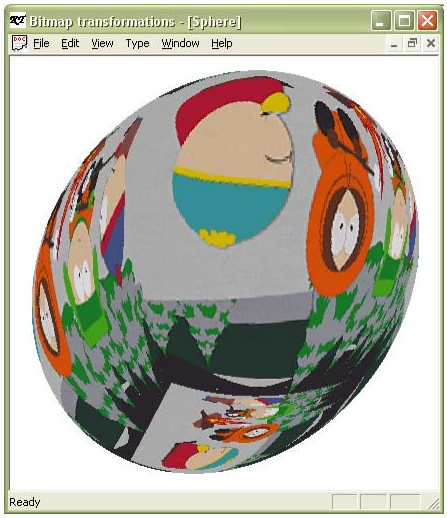


Figure 3.5: Six bitmaps transformed to a sphere

3.3 Methods for Free-form Patches

When we take a non-symmetrical shape and do not want a break in the pattern, then we shall seek new methods. The following algorithms try to widen the shapes that can be handled.

3.3.1 Shrinking the Borders

In the sequel we suppose that the bitmap to be transformed is $N \times N$. It is not a restriction in practice, because each rectangular bitmap easily can be transformed into a $N \times N$ shape. Let the transformed borders given by $(x_1(t_1), y_1(t_1))$, $(x_2(t_2), y_2(t_2))$, $(x_3(t_3), y_3(t_3))$, $(x_4(t_4), y_4(t_4))$ function pairs, where $t_1, t_2, t_3, t_4 \in [0, 1]$.

The points of intersections of the border curves defined by function pairs $(x_i(t_i), y_i(t_i))$ $(i = 1, 2, 3, 4)$ are the followings: $(x_1(1), y_1(1)) = (x_2(0), y_2(0))$, $(x_2(1), y_2(1)) = (x_3(1), y_3(1))$, $(x_1(0), y_1(0)) = (x_4(0), y_4(0))$ and $(x_4(1),$

$y_4(1)) = (x_3(0),\ y_3(0))$. Let $f_i(t_i) = (x_i(t_i),\ y_i(t_i))$ and $a_i(t_i, s) = f_i(t_i) + (KP - f_i(t_i)) \cdot s/(N/2)$, $i = 1, 2, 3, 4$. Let KP denote the point of intersection of the diameters defined by the transformed vertices of the original grid. The diameters divide the form into four parts. The following constraints have to be satisfied for each $f_i(t_i)$: neither $f_i(t_i)$ can have a point of intersection with the diameters and each $f_i(t_i)$ must has one and only one point of intersection with those lines which fit on KP and lie in the appropriate angular region. Changing any of the $f_i(t_i)$ will only affect the appropriate quarter of the image.

The transformed edge points of the original grid are the followings:

$$\begin{array}{lllll}
(x_1(0), y_1(0)) & (x_1(1/N), y_1(1/N)) & (x_1(2/N), y_1(2/N)) & \dots & (x_1(1), y_1(1)); \\
(x_2(0), y_2(0)) & (x_2(1/N), y_2(1/N)) & (x_2(2/N), y_2(2/N)) & \dots & (x_2(1), y_2(1)); \\
(x_3(0), y_3(0)) & (x_3(1/N), y_3(1/N)) & (x_3(2/N), y_3(2/N)) & \dots & (x_3(1), y_3(1)); \\
(x_4(0), y_4(0)) & (x_4(1/N), y_4(1/N)) & (x_4(2/N), y_4(2/N)) & \dots & (x_4(1), y_4(1)).
\end{array}$$

The transformed inner grid points can be obtained by giving the parameters of functions a_i. These points will be defined from the edge moving toward KP. In each step less and less points by two have to be defined.

The parameters for a quarter are:

$$\begin{array}{lllll}
a_i(0,0) & a_i(1/N,0) & a_i(2/N,0) & \dots & a_i(1,0); \\
a_i(0,1/(N/2)) & a_i(1/(N-2),1/(N/2)) & a_i(2/(N-2),1/(N/2)) & \dots & a_i(1,1/(N/2)); \\
a_i(0,2/(N/2)) & a_i(1/(N-4),2/(N/2)) & a_i(2/(N-4),2/(N/2)) & \dots & a_i(1,2/(N/2));
\end{array}$$

$$\dots$$

$$\begin{array}{lll}
a_i(0,(N/2-1)/(N/2)) & a_i(1/2,(N/2-1)/(N/2)) & a_i(1,(N/2-1)/(N/2)); \\
\multicolumn{3}{c}{a_i(1,1).}
\end{array}$$

The common points of the neighbouring quarters will be computed twice and the point of intersection of the diameters will be computed four times in this way. Naturally, it is enough to compute these points only once.

The advantage of this method is that the transformed image can be modified locally. Changing only one bordering curve implies the recalculation and redrawing of only one quarter of the image. The four quarters of the image are distorted according to the bordering curves. The quarters are independent from each other, but the projection is continuous at the common edge of the quarters (see Figure 3.6).

3.3.2 Using Bilinear Weight Function

Transforming to a free-form patch can be interpreted the following way: two function sequences have to be defined between two pair of functions and we have to use the points of intersections of these function sequences (see [27]). Let the original bitmap be $N \times M$.

Figure 3.6: The original bitmap transformed to a free-form patch

Let $a_1(u)$, $a_2(u)$, $u \in [0,1]$, és $b_1(v)$, $b_2(v)$, $v \in [0,1]$ curves be given so that they are in the same plane and both a curve have one point of intersection with both b curves. These curve pairs will define the transformed bitmap. Our objective is to find a $s(u,v)$, $u,v \in [0,1]$ function for which $s(u,0) = a_1(u)$, $s(u,1) = a_2(u)$; $s(0,v) = b_1(v)$, $s(1,v) = b_2(v)$ are satisfied.

We will use function sequences to solve this problem. Let us consider $I_a(u,v) = (1-v) \cdot a_1(u) + v \cdot a_2(u)$ function sequence defined by $a_1(u)$ and $a_2(u)$ and let us consider $I_b(u,v) = (1-u) \cdot b_1(v) + u \cdot b_2(v)$ function sequence defined by $b_1(v)$ and $b_2(v)$. These function sequences are interpolating the opposite curves but their end points do not follow the other curve pair. The bilinear interpolation of the four vertices will solve this problem.

$$I_{ab}(u,v) = \begin{bmatrix} 1-u & u \end{bmatrix} \cdot \begin{bmatrix} s(0,0) & s(0,1) \\ s(1,0) & s(1,1) \end{bmatrix} \cdot \begin{bmatrix} 1-v \\ v \end{bmatrix}$$

The $s(u,v) = I_a(u,v) + I_b(u,v) - I_{ab}(u,v)$ function yields the solution of the problem (see Figure 3.7). The transformed grid can be obtained by using the following parameters:

$$\begin{bmatrix} s(0,0) & s(0,1/M) & s(0,2/M) & \dots & s(0,1) \\ s(1/N,0) & s(1/N,1/M) & s(1/N,2/M) & \dots & s(1/N,1) \\ & & & \dots & \\ s((N-1)/N,0) & s((N-1)/N,1/M) & s((N-1)/N,2/M) & \dots & s((N-1)/N,1) \\ s(1,0) & s(1,1/M) & s(1,2/M) & \dots & s(1,1) \end{bmatrix}.$$

The $s(u,v)$ function can yield such co-ordinate pair for inner points that defines a point that is out of the area defined by the four curves. We have to take into consideration this fact at giving the bordering curves.

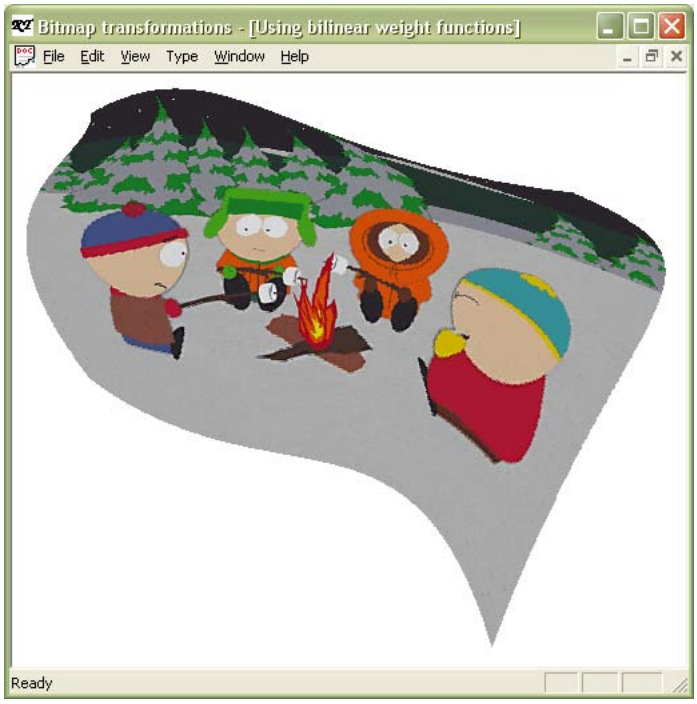


Figure 3.7: Transformation based on bilinear weight function

Changing the borders effects the whole picture, so we have to recalculate all of the points of the transformed grid. In spite of this fact, we can speed up the algorithm by reusing partial results.

3.4 Using Neural Networks

This section will present a new method based on neural networks which performs bitmap transformation onto a free-form patch. An important advantage of this new method that arbitrary topology can be chosen for a pixel (as might as well hexagon or triangular instead of the usual quadrilateral), because this topology remains fixed during the algorithm. On the other hand, it has a big disadvantage considering the solutions so far: this method is rather slow, thus making animations is not possible.

3.4.1 Introduction into Neural Networks

The theorem of artificial neural networks yields a very effective alternative method (see [28], [29]). In the followings a Kohonen (see [30], [31], [32], [33]) network will be used because of its strong ability to learn data.

The approximation algorithm consists of two separated parts.

1. The neural network is taught by random points and by this teaching procedure prepares a grid of a chosen topology. The starting grid points will spread over the free-form patch.

2. The areas corresponding to the pixels have to be filled up with the appropriate color.

3.4.2 The Training Procedure

The network will be trained by points selected from a uniformly distributed area, see [30]. In order to obtain the desired form, we put the points of the border in regularly. Since our problem is two-dimensional, the input layer consists of two nodes. The output layer consists of $m = (k+1) \cdot (l+1)$ where k and l are the number of the rows and columns of the original bitmap. These m output nodes form a grid. Each nodes of the input layer is connected to each nodes of the output layer. All of the connections have a weight: w_{ij} denotes the weight between the input node i and output node j.

- Coordinates of the input points are: $p_i(x_{1i}, x_{2i})$ $(i = 1, \ldots, n)$, where n is the number of input vectors.

- Coordinates of the output points are: $q_j(w_{1j}, w_{2j})$ $(j = 1, \ldots, m)$, where m is the number of the nodes in the network.

The training procedure is defined in the following way:

Step 1. Initialize the weights w_{sj}, $(s = 1, 2; j = 1, \ldots, m)$ according to the chosen topology. Let the training time $t = 1$.

Step 2. Present new input values (x_{1i_0}, x_{2i_0}), as the coordinates of a randomly selected input point p_{i_0} from the free-form patch. We take a point from the border regularly, too.

Step 3. Compute the Euclidean distance of all output nodes to the input point:

$$d_j = \sum_{s=1}^{2} (x_{si_0} - w_{sj})^2.$$

Step 4. Find the winning unit q_{j_0} as the node which has the minimum distance to the input point, so where j_0 is the value for which $d_{j_0} = min(d_j)$.

Step 5. Compute the neighborhood $N(t) = (j_0, j_1, \ldots, j_k)$.

Step 6. Update the weights of the nodes in the neighbourhood by the following equation:

$$w_{sj}(t+1) = w_{sj}(t) + \eta(t) \cdot (x_{si_0} - w_{sj}(t))^2 \quad \forall j \in N(t),$$

where $\eta(t)$ is a gain term decreasing in time.

Step 7. Let $t = t + 1$. Repeat Step 2-7 until the network is trained. This means that the change of the grid is under a predefined limit.

Initializing the Weights

They have to be determined in a way that the starting grid points form a rectangle within the free-form patch. Best results can be achieved by arranging the weights around the centroid of the patch.

The Radius and the Gain Terms

Both of them have to decrease in time and they shall be Gaussian. In the algorithm I have used the following formulas to calculate the radius and the gain term, see [32] and [33]:

$$Gain(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{t}{q})^2} \quad Radius(t) = \frac{n}{2} e^{-\frac{1}{2}(\frac{t}{s})^2},$$

where t denotes the number of iterations, q and s are the scales on the time axis, while n denotes the number of input points.

By using artificial neural networks we get very nice transformed pictures, but unfortunately the speed is not satisfactory. The pixels from which the picture is made up have similar size if the parameters are chosen expediently. This method helps us to extend the set of free-forms which can be textured.

References

[1] J. Dassoa, J. Leeb, H. Bacha and R. G. Mage: *A comparison of ELISA and flow microsphere-based assays for quantification of immunoglobulins.* Journal of Immunological Methods, Volume 263, Issues 1-2, 1 May 2002, Pages 23-33

[2] T. Lilliehorn, M. Nilsson, U. Simu, S. Johansson, M. Almqvist, J. Nilsson, T. Laurell: *Dynamic arraying of microbeads for bioassays in microfluidic channels.* Sensors and Actuators B-Chemical 106 (2): May 13 2005, Pages 851-858

[3] L. A. Ugozzoli: *Multiplex assays with fluorescent microbead readout: A powerful tool for mutation detection.* Clinical Chemistry 50 (11): Nov 2004, Pages 1963-1965

[4] V. Moalic, B. Mercier and C. Ferec: *Technologie Luminex^{TM} : principe, applications, et perspectives; Luminex^{TM} Technology: technical approach, applications and future prospects.* Immuno-analyse & Biologie Spécialisée, Volume 19, Issue 4, August 2004, Pages 181-187

[5] B. L. Ray, D. Ebner and I. Balazs: *Identifying HLA antibodies using luminex microspheres.* Human Immunology, Volume 63, Issue 10, Supplement 1, October 2002, Page S85

[6] Dario A. A. Vignali: *Multiplexed particle-based flow cytometric assays.* Journal of Immunological Methods, Volume 243, Issues 1-2, 21 September 2000, Pages 243-255, Flow Cytometry

[7] http://www.cytometry.com

[8] http://www.appliedcytometry.us

[9] *Human Interface Guide* http://developer.apple.com/documentation/ /UserExperience/Conceptual/OSXHIGuidelines/index.html

[10] Soft Flow Kft. http://www.softflow.com

[11] Dr. Szirmay-Kalos László, Márton Gábor, Dobos Balázs, Horváth Tamás, Risztics Péter, Kovács Endre: *Theory of Three Dimensional Computer Graphics (PDF version).* http://www.iit.bme.hu/ szirmay/book.html

[12] Dr. Szirmay-Kalos László: *Számítógépes grafika.* Computerbooks, Budapest, 2001

[13] Dr. Szirmay-Kalos László, Antal György, Csonka Ferenc: *Háromdimenziós grafika, animáció és játékfejlesztés.* Computerbooks, Budapest, 2003

[14] E. Angel: *Computer Graphics.* Addison-Wesley Publishing Co., 1990

[15] A. Watt: *3D Computer Graphics.* Addison-Wesley, 1993

[16] Bácsó S., Papp I., Szabó J.: *Projektív geometria.* Mobidiák rendszer, http://mobidiak.inf.unideb.hu, Debrecen, 2004

[17] Hajós György: *Bevezetés a geometriába.* Tankönyvkiadó vállalat, Budapest, 1971

[18] Reiman István: *A geometria és határterületei.* Gondolat, Budapest, 1986

[19] F. P. Preparata, M. I. Shamos: *Computational Geometry.* Springer-Verlag, 1985

[20] Varga Ottó: *Projektív geometria.* Felsőoktatási Jegyzetellátó Vállalat, 1954

[21] I. Herman: *The Use of Projective Geometry in Computer Graphics.* Springer Verlag, Berlin, 1992

[22] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes: *Computer Graphics Principles and Practice.* Addison-Wesley, New York, 1992

[23] D. F. Rogers, J. A. Adams: *Mathematical Elements for Computer Graphics.* McGraw-Hill, 1990

[24] M. O'Rourke: *Principles of Three Dimensional Computer Animation (Modeling, Rendering & Animating with 3D Computer Graphics).* W. W. Norton & Company, New York, 1998

[25] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes and R. L. Phillips: *Introduction to Computer Graphics.* Addison-Wesley Publishing Co., 1994

[26] G. Moore: *Cramming more components onto integrated circuits.* Electronics, 38, 8, 1965

[27] Juhász Imre: *Számítógépi geometria és grafika.* Miskolci egyetemi kiadó, Miskolc, 1995

[28] J. Freeman, D. Skapura: *Neural Networks; Algorithms, Applications and Programming Techniques.* Addison-Wesley, 1991

[29] R. Rojas: *Neural Networks. A Systematic Introduction.* Springer-Verlag, 1996

[30] T. Kohonen: *Self-organization and associative memory. 3rd edition.* Springer-Verlag, 1989

[31] M. Alder, R. Togneri, E. Lai, Y. Attikiouzel: *Kohonen's algorithm for the numerical parametrisation of manifolds.* Pattern Recognition Letters 11, 1990, Pages 313-319

[32] M. Hoffmann: *Modified Kohonen Network for Surface Reconstruction.* Publ. Math., Debrecen, 1998

[33] M. Hoffmann, L. Várady: *Free-form Surfaces for Scattered Data by Neural Networks.* Journal for Geometry and Graphics, 1998

List of Talks

1. *Bitmap transzformációk projektív invariánsok alapján*, TDK Conference, Debrecen, 1996.
2. *Bitmap transzformációk projektív invariánsok alapján*, OTDK Conference, Szeged, 1997.
3. *Bitmap Transformations Based on Projective Invariants*, Conference on Computer Graphics, Wisla, 1997.
4. *Bitmap Transformations*, 4^{th} International Conference on Applied Informatics '99, Noszvaj, 1999.
5. *Bitmap Transformations*, lecture at seminar of Technical University of Wien, Wien, 2000.
6. *Bitmap Transformations by Neural Networks*, 5^{th} International Conference on Applied Informatics '01, Eger, 2001.
7. *3D Visualization for Mobile Equipments*, Conference on Computer Graphics, Szyrzk, 2003.
8. *3D visualization for mobile phones*, II. Hungarian Conference on Computer Graphics and Geometry, Budapest, 2003.
9. *MMA Technology*, 6^{th} International Conference on Applied Informatics '04, Eger, 2004.
10. *MMA Technology and Cross-platform Developing*, CSCS Conference, Szeged, 2004.

List of Publications and Teaching Materials

Reviewed Publications

1. CS 0997.68168MA Schwarcz, Tibor; Tornai, Róbert: *Bitmap transformations. (English)*
   Kovács, Emöd (ed.) et al., Proceedings of the 4th international conference on applied informatics. Education and other fields of applied informatics, computer graphics, computer statistics and modeling. Eger-Noszvaj, Hungary, August 30-September 3,
   1999. Eger: Molnár és Társa, 169-178 (2001). CR: *G.4 I.3

2. MR2125603 Tornai, Róbert(H-LAJO-II): *Shape modification of cubic B-spline curves by means of knot pairs. (English. English summary)*
   Acta Acad. Paedagog. Agriensis Sect. Mat.
   (N.S.) 31 (2004), 61–68. 65D17 (68U05)

Publications

1. Schwarcz Tibor, Tornai Róbert: *Bitmap transformation based on projective invariants*
   GEOMETRIA I GRAFIKA INŻYNIERSKA z. 2
   15-23, Gliwice 1998

2. Tornai Róbert: *Free-form Bitmap Transformations by Neural Networks*
   5^{th} International Conference on Applied Informatics '01
   konferenciakiadvány, 229-234, Eger 2001 (under review)

3. Tornai Róbert: *3D visualization for mobile phones*
   II. Magyar Számítógépes Grafika és Geometria Konferencia
   konferenciakiadvány, 129-134, Budapest 2003, ISBN 963 420 766 9

4. Tornai Róbert: *3D graphics for mobile equipments*
   GEOMETRIA I GRAFIKA INŻYNIERSKA z. 6
   53-59, Gliwice 2004

5. Tornai Róbert: *MMA Technology*
   6^{th} International Conference on Applied Informatics '04
   konferenciakiadvány, 449-456, Eger 2004 (under review)

6. É. Széles, B. Kovács, R. Tornai, Z. Győri, H. Berndt: *HHPN for sample introduction in Flame AAS using subcritical CO_2 as liquid/gas pressure pump*
   4^{th} Aegean Analytical Chemistry Days
   29 September - 3 October, Kusadasi/Aydin - Turkey,
   Proceedings book, p.: 584-586
   Abstracts book, p.: 147

Publications Joint with the Dissertation

1. Schwarcz Tibor, Tornai Róbert: *Bitmap transformation based on projective invariants*
   GEOMETRIA I GRAFIKA INŻYNIERSKA z. 2
   15-23, Gliwice 1998

2. CS 0997.68168MA Schwarcz, Tibor; Tornai, Róbert: *Bitmap transformations. (English)*
   Kovács, Emöd (ed.) et al., Proceedings of the 4th international conference on applied informatics. Education and other fields of applied informatics, computer graphics, computer statistics and modeling. Eger-Noszvaj, Hungary, August 30-September 3,
   1999. Eger: Molnár és Társa, 169-178 (2001). CR: *G.4 I.3

3. Tornai Róbert: *Free-form Bitmap Transformations by Neural Networks*
   5^{th} International Conference on Applied Informatics '01
   konferenciakiadvány, 229-234, Eger 2001 (under review)

4. Tornai Róbert: *MMA Technology*
   6^{th} International Conference on Applied Informatics '04
   konferenciakiadvány, 449-456, Eger 2004 (under review)

5. MR2125603 Tornai, Róbert(H-LAJO-II): *Shape modification of cubic B-spline curves by means of knot pairs. (English. English summary)*
   Acta Acad. Paedagog. Agriensis Sect. Mat.
   (N.S.) 31 (2004), 61–68. 65D17 (68U05)

Teaching Material

1. Tornai Róbert:
   Fejezetek a számítógépi grafikából (lektorált egyetemi segédanyag)
   85 oldal, 2003, Mobidiák rendszer, http://mobidiak.inf.unideb.hu