Osvrt na predavanje

Na početku predavanja profesor Pap govori o programu Fotographeru u kojemu možemo sami izrađivati fontove. Njegova alternativa je program Fontlab. Font je uređena nakupina kodnih pozicija, a na svakoj kodnoj poziciji imamo sliku (eng. glyph). U programu font tablicu možemo gledati kroz klasičnu ASCII kodnu tablicu koja je napravljena prema američkom standardu koji je određivao kako će iglice u matričnom printeru podesiti za prikaz određenog znaka. Osim decimalnog, postoji i koordinatni sustav slova gdje pravci stvaraju zatvoreni koordinatni sustav koji je omeđen jednadžbama pravaca i kojeg nazivamo digitalni četverac. Ti pravci su beskonačni te nam omogućuju da sami dizajniramo slova. Slaganje slova jedno s drugima možemo testirati u raznim programima kao što su: Photoshop, Indesign ili Ilustrator. Međutim, slova tijekom izrade možemo ispitati i u samom Fontographu u kojem ćemo za tu svrhu otvoriti prozor „Metrics“. Na primjeru slova A i V možemo vidjeti preveliku udaljenost između slova zbog koje se riječ „AVA“ doima nečitljivom. Ta udaljenost je urokovana međusobnim preklapanjem rubova digitalnih četveraca koje, u ovom slučaju, nije poželjno. Radi takvih slučajeva radimo iznimke, a to su tzv. parovi podrezivanja. Slova pomičemo bliže pomoću linije označene sa slovom K, a u donjoj tablici možemo pratiti koliko smo pomaknuli koje slovo i u kojem smjeru. Kada popravimo takve slučajeve unutar fonta to ostane zapamćeno te čim taj font selektiramo u nekom drugom programu kao što je primjerice Word onda će se uključiti i parovi podrezivanja tijekom pisanja teksta. Prilikom dizajniranja fonta bitno je dobro poznavanje slova. Tako da ćemo uvijek prvo dizajnirati npr. slovo C pa onda na njega dodati kvačicu kako bi dobili Č. Ako želimo napraviti rukopisni font onda prvo moramo ispisati što više slova, potom izabrati ono koje nam se čini tipično za vlastiti rukopis, skenirati ga i staviti u kodnu tablicu te aktivirati vektorizaciju slike. Tako postignemo da program automatski pretvori cijelu sliku po rubovima u Bezierove krivulje. PSConvert je softver napisan u programskom jeziku C++. Ima jednostavnu korisničku masku koja prikazuje tipične grafičke parametre (rezolucija, širina, visina) i još tri parametra: prvo slovo, drugo slovo i gustoća. Kada pokrenemo softver on stvara sliku koju otvaramo u Photoshopu jer softver generira tif zapis. Ovaj program dizajnira linije koje se kao lepeza šire iz jedne točke od 0 do 180 stupnjeva, te zato program stvara maske i određeni koordinatni sustav kako bi mogli pozicionirati određene slike. Ta lepeza linija je vidljiva samo kroz masku slova koje smo upisali kao parametre u PSConvert. Ovaj je program u usporedbi s Illustratorom puno brži. PSTipograf ima simulator koji ima puno više parametra od prethodnog softvera; prozor za opis teksta koji želimo staviti u krug, rezolucija, visina i širina. Možemo definirati i visinu fonta, x i y koordinate početka teksta, hue te faktor snage spirale. Prvo reguliramo kako će izgledati ispis i tako shvaćamo da svaki parametar utječe ne samo na grafiku nego i na okvir, tj. digitalno platno na koje će ta grafika doći. Bezierove krivulje su označene s četiri točke; prva točka, dvije natezne ili tangentne točke i završnu točku. Pomoćne ili tangentne točke krivulje

su označene s dva plusa. To je matematički način rada. U svim softverima se radi preko povezanih točaka jer je jedan plus preko spojne točke funkcijski povezan s drugim plusom i to bi bila jednadžba kroz tri točke. Ako spojnu točku prebacimo u drugi način rada „corner point“ onda je nezavisna, tj. nema više jednadžbe između dva plusa. Ako istu točku prebacimo u treći način spajanja, „tangent point“, onda spojna točka postaje tangenta na krivulju. Stoga je bitno da poznajemo režim spajanja Bezierovih točaka, a ne baviti se samo čistom upotrebom. Bezierove krivulje su parametarske krivulje trećeg stupnja, iz skupine predvidljivih krivulja kod kojih se pomoću položaja kontrolnih točaka, koje su u domeni rada krivulje, radi predikcija gdje bi te krivulje stvarno trebale ići. Prvi su se put počele upotrebljavati za dizajn haube u tvornici automobila, a danas se upotrebljava u svim programima gdje se mora raditi vektorska staza. Kada se nauči teorija onda je jednostavno otići u postscript i sami naredbama to napraviti. Moramo imati postscript drivere koji rade konverziju iz svih jezika u jezik poznat ispisnoj tehnologiji kako bi mogli pravilno spremiti svoje radove. Ghostscript je simulacija ispisa, tj. to je program koji nam može prikazati bilo koji postscript kod koji mu pošaljemo. Možemo birati hoće li ispis biti crno-bijel ili u boji. „Curve to“ je naredba za Bezierovu krivulju. Sadrži 6 brojeva umjesto 8 jer je ta naredba mikrokodom definirana tako da prvu točku uzima kao momentalno tekuću radnu točku postscripta koju moramo stvoriti prije zadavanja naredbe. Zato prije korištenja te naredbe ide naredba „move to“. Bezierovu krivulju možemo koristiti i u drugim tehnologijama i jezicima, primjerice u vektorskoj grafici za web. Jedan od standardnih današnjih tehnologija je SVG jezik kojeg poznaju svi browseri. Taj jezik u sebi ima jako slične naredbe kao postscript, tako da kad se on nauči kao glavni jezik vektorske grafike, nije problem prijeći i na neki drugi jezik koji radi vektorsku grafiku. SVG (Scalable Vector Graphics) je jezik iz obitelji XML jezika. Vektorska grafika nije vezana za rezoluciju već je vezana samo za trenutak ispisa kada nešto prikazujemo. Krivulja će, dakle, ostati glatka i kad promijenimo rezoluciju. Suprotno tome, u piksel grafici, koju npr. stvaraju konstruktori slike u Photoshopu, rezolucija je zadana i ne možemo ju skalirat, već ju možemo samo semplirati što zamućuje sliku. Na primjeru imamo krivuljastu stazu i jedan trokut koji putuje po toj stazi. Ako želimo da je staza vidljiva „M“ označava „move to“ a „C“ označava „curve to“. „Move to“ ima dvije brojke, a „curve to“ njih šest i tako je u svakom programu. Također imamo i opciju „style“ kojom možemo odrediti hoće li staza imati ispunu ili ne, te koje će boje biti. Ako ne želimo vidjeti stazu, onda naredbu za stazu ogradimo. Postoji i naredba za određivanje brzine kojom se trokut kreće po stazi. Dinamičku rastersku podlogu dobijemo pomoću rastriranja (dobivanje različitih nijansa jedne boje). U primjeru na podlozi možemo vidjeti sive razine koje se dobro vide na ekranu, ali se u tisku jako teško dobivaju, osim ako imamo EF rastriranje. Udaljenost i veličina rasterskih elemenata određuje gdje će nijansa biti tamnija ili svjetlija. Pri amplitudnom modeliranju je stalna frekvencija udaljenosti, ali se simulacija sivoće postiže povećanim ili smanjenim rasterskim elementom. U prikazu vidimo egzotične rasterske elemente, a ne standardne, kao primjerice točkice. Podloga je pod regulacijom određene formule koja poziva puno naredbi. Nema piksela već rasterski elementi koji se fizički tiskaju u tisku s određenom gustoćom nama simuliraju sivoću. Kako bi doživjeli suhoparne

matematičke formule onda ih trebamo zapisati u nekom jeziku određenom sintaksom s naredbama add, sub, div, idiv i druge. Prvo treba vizualizirati što želimo dobiti, a za to postoji eksperimentacijski notebook koji ima puno već priređenih formula da bi se vidio mehanizam, a kasnije kada se navikne na mehanizam se mijenjaju formule. U primjeru se koriste naredbe „Plot3D“ i „ContourPlot“ iz matematike te se koristi poziv z funkcije. Da bi se programski koristile boje treba ih dobro poznavati i ograničenja obojenja u CMY i RGB sustavu. Potrebno je inženjerski pristupiti odabiru boja jer ako, primjerice, otisnemo boju na boju, transparentnu i pokrivnu, onda ćemo ako je transparentna u oku percipirati neku treću boju, dok ćemo u slučaju pokrivnih dobiti onu prvu boju. U HTML-u smo ograničeni na RGB sustav boja koji ne postoji u ispisnom sustavu jer ne sadrži crvenu, plavu i zelenu bočicu boja. To je napravljeno u ispisu pomoću CMY sustava boja. Kako bi upotrijebili neku boju u postscriptu možemo se igrati s različitim color sustavima. U primjeru imamo kvadrat koji je obojan u HSB sustavu. Mijenjamo ga u CMYK sustav i onda za to koristimo četiri parametra. Ulazimo u Javascript, mijenjamo boju pozadine te odlučujemo hoće li program pristupiti po HTML standardu s imenima boja ili s kodom iz RGB-a. Može se upotrijebiti više tehnologija za prikazivanje RGB i CMYK sustava boja. Word poznaje samo RGB sustav, a ako u njega ubacimo CMYK tiff i ispišemo onda dolazi do konverzije CMYK tiffa u RGB i onda iz RGB na printer te naposljetku to više ne bude ona slika koju smo napravili u Photoshopu. Oboje HTML i PDF prikazuju tekst, slike i ostalo, ali PDF može biti prikazan i u CMYK sustavu , dok HTML samo u RGB sustavu. Najvažnije je to što PDF poznaje pojam stranice (page), a HTML ne i zbog toga moramo imati drugačiji pristup svakom od softvera. Postoji XML jezik za kontrolu PDF-a. U PDF-u imamo naredbe koje reguliraju donju, gornju, lijevu, desnu, te središnju marginu ako postoji. To se sve radi i u najmodernijim browserima koji imaju određen plug in. U PDF-Uu postoji destilator ima puno parametara, a služi da iz postscripta stvara PDF i to možemo kontrolirati na koji način, tj. možemo destilirati to na svoj način. A možemo to napraviti s naredbama, možemo kontrolirati numeriranje (desne stranice neparne, lijeve parne).