OBSAH

1. ÚVOD
   1. O čem ta práce je, seznamení
2. Technologie
   1. HTML5
   2. WEBGL:
      1. jakou má funkci,
      2. z čeho vychází,
      3. proč vůbec existuje
      4. k čemu se nejčastěji pouziva
3. Shadery
   1. GLSL
   2. Vertex shader
   3. Fragment shader
4. Návrh řešení
   1. Popis kamery
   2. Postup běhu programu
      1. 1.) načtení a vytvoření WebGL CANVASu
      2. 2.) vytvoření a kompilace VERTEX SHADERU
      3. 3.) vytvoření a kompilace FRAGMENT SHADERU
      4. 4.) vytvoření programu
      5. 5.) vytvoření a načtení do BUFFERU
5. Implementace
   1. Tvorba shaderů
   2. Tvorba geometrie
   3. Buffery
      1. Vertex => [x,y,z]
      2. Normal => [x,y,z] (data bez radiusu)
      3. Texture data => [u,v]
      4. Index data => [w,e,r,t,h,m]
   4. Nacteni textur videa
   5. Transformační matice a operace
   6. Ovládání prohlížeče
   7. Kompas
   8. Jak bylo vyřešeno bezešvé prohlížení
      1. Gauss / pyramida, blending obecně
6. Závěr
7. Literatura
8. Přílohy

Geometrie – Vlastní práce

Aby panoramatický prohlížeč jednotlivých fotek či videí dokázal vyobrazit naše data, je nejprve potřeba vytvořit geometrii. Jedná se o pole souřadnic formátu [x1,y1,z1 , x2,y2,z2,…] , které utváří jednotlivé body v prostoru, body pak mezi sebou tvoří prostor, na který budou data mapovaná. Ve Webgl umíme vykreslit jednolitlivé body, čáry – tedy spojnice jednotlivých bodů , nebo trojúhelníky. V našem případě bude potřeba vykreslit texturu jako plochu, na kterou se bude vše mapovat. Pro ten to případ se ve webgl používá vykreslení pomocí trojúhelníku, jako základní jednotkou prostoru. Způsob vykreslení našich bodu určíme tedy řádkem: gl.drawElements(gl.TRINAGLES, 6, gl.UNSIGNED\_BYTE, 0), kde tedy říkáme že chceme vykreslovat plochu pomocí trojúhelníků, víz výše. (gl.TRINAGLES), druhým parametrem říkáme, kolik hodnot z index bufferu budeme potřebovat vykreslit, třetím parametrem oznamujeme velikost jednoho indexu a posledním parametrem říkáme, odkud začíná program číst naše body. Mohli bychom naše body vykreslovat ve webgl i pomoci příkazu: gl.drawArrays(), ale došlo by pak k situaci, že by se nám již vykreslené body zbytečně vykreslovali znovu. Abychom předešli duplikaci při vykreslování, bude nutné si vytvořit pole indexů, které nám budou ukazovat na jednotlivé vrcholy geometrie. Zde se již mohou indexy opakovat.

…

Geometrie prohlížeče je nastavena na 60 vertikálních spojnic severního polu s a jižním polem. Horizontálních čar (rovnoběžek) budeme potřebovat o něco méně, takže je nastavíme na hodnotu 50. Čím více rovnoběžek a poledníků vytvořili, tím by byla hustota bodů v polích hustší, a mapování textury tedy jemnější. Tato vlastnost se nám bude hodit např. v situaci, kdy bychom měli velký geometrický objekt se spoustou miniaturních sekvenci v geometrii. Větší mezery jsou nadruhou stranu rychlejší na vykreslování a to se stejným výsledkem (v našem konkrétním případě). Geometrii nastavíme výpočtem souřadnic X, Y, Z. Jelikož poledník je rozdělen rovnoběžkami, můžeme tedy 180° úhel rozdělit mezi jednotlivé rovnoběžky, tedy PÍ/Počet rovnoběžek = theta, který bude svírat vertikální čára geometrie s osou Z, tudíž můžeme souřadnici X vyjádřit vstahem X= sin(theta) \* cos(phi). Vypočet bodů