

Příprava na 4. cvičení

Na čtvrtém cvičení budeme probírat deferred shading. Pro hladký průběh cvičení si zopakujte a připravte:

- Zopakujte si Phongův osvětlovací model, zejména jak vyhodnotit osvětlení, máte-li pozici, normálu a barvu materiálu (ne nutně z vertex shaderu).
- Zopakujte si framebuffer, fragment shader bude mít na výstupu tři proměnné.
- Zopakujte si, k čemu slouží funkce *glDepthMask* a jak se používá.
- Zopakujte si míchání barev (blending), funkci *glBlendFunc* a její parametry (*GL_ONE*, *GL_ZERO*, *GL_SRC_ALPHA*, *GL_ONE_MINUS_SRC_ALPHA*).
- Zopakujte si, k čemu slouží v GLSL *discard* a jak se používá.
- Zopakujte si instancování, budeme instancovaně kreslit koule.

Aktualizujte si Framework.zip:

- Byla aktualizována kamera, nyní lze nastavit bod, okolo kterého se kamera otáčí (metoda *SetCenter*).
- Byla aktualizována třída *PhongLightsData_UBO*, nyní lze zadat, jestli mají být data v UBO nebo v SSBO (shader storage buffer object, co to je a jak se to liší od UBO se dozvíte na přednášce).

Projděte si projekt Cv4 ve studijních materiálech. Zaměřte se zejména na:

- Máme náhodně vygenerovanou scénu, objekty i světla. Kód generující scénu si můžete projít, ale nemusíte se na něj zaměřovat, upravovat to nebudeme. Všimněte si, že scéna obsahuje až 150 světél, která jsou bodová a mají útlum (attenuation).
- Máme 2 framebuffer objekty. Všimněte si zejména framebuffer objektu *GbufferFBO*, který má tři textury navázané na tři color attachmenty, takže můžeme z fragment shaderu dávat na výstup tři proměnné, do každé textury jednu. Také si všimněte, že oba FBO mají textury stejně velké jako je okno, do kterého kreslíme, takže je nutné při každé změně velikosti okna aktualizovat velikost těchto textur.
- Projděte si část kódu v *render_scene* pro *if USE_FORWARD_SHADING* a shadery *notexture_vertex.glsl*, *texture_vertex.glsl*, *notexture_forward_fragment.glsl* a *texture_forward_fragment.glsl*. Je to kód, který kreslí naši scénu tak, jak jsme byli doposud zvyklí, měl by vám být jasný.
- Shadery *fullscreen_quad_vertex.glsl* a *display_texture_fragment.glsl* slouží k zobrazení textur na obrazovku, měly by vám být jasné.

- Kód by měl jít spustit, na počátku máme scénu s až 150 světly, kterou kreslíme tak, jak jsme byli doposud zvyklí. Všimněte si doby kreslení jednoho snímku, která se zobrazuje v GUI.

Ve Cv4 jsou navíc tyto věci, které ještě nebyly probrány a které budou probrány na přednášce:

- Shader storage buffer objekty (*GL_SHADER_STORAGE_BUFFER* v C++ a *buffer* v GLSL)
- Kvalifikátory *flat* a *noperspective* v *evaluate_sphere_vertex.glsl* a v *evaluate_sphere_fragment.glsl*
- OpenGL query pro měření doby výpočtu (proměnná *RenderTimeQuery* a věci okolo ní)