Dokumentacja pliku `bump_map_demo.py`

Poniższy dokument opisuje moduły, funkcje i klasy zastosowane w pliku `bump_map_demo.py`. Celem programu jest demonstracja bump mappingu (mapowania wypukłości) w silniku graficznym Panda3D.

---

1. Moduły i zależności

```
from direct.showbase.ShowBase import ShowBase
from panda3d.core import loadPrcFileData, WindowProperties, Filename, Shader, AmbientLight, PointLight, TextNode, LPoint3, LVector3
from direct.task.Task import Task
from direct.actor.Actor import Actor
from direct.gui.OnscreenText import OnscreenText
from direct.showbase.DirectObject import DirectObject
from direct.filter.CommonFilters import *
import sys
import os
```

- direct.showbase.ShowBase.ShowBase: Klasa bazowa aplikacji Panda3D, umożliwia tworzenie okna renderującego, obsługę zdarzeń i zasobów.
- panda3d.core:
  - loadPrcFileData: umożliwia dynamiczne ustawianie parametrów konfiguracyjnych

silnika.

  - WindowProperties: pozwala na konfigurację właściwości okna (np. ukrycie kursora).

  - Filename: klasa reprezentująca ścieżki plików wewnątrz Panda3D.

  - Shader: klasa reprezentująca shader GLSL.

  - AmbientLight i PointLight: klasy oświetlenia scenicznego.

  - TextNode: tworzenie tekstów w przestrzeni 3D.

  - LPoint3, LVector3: wektory punktów i wektorów 3D.

- direct.task.Task.Task: mechanizm systemu zadań (task manager), wywołuje funkcje w każdej klatce.

- direct.actor.Actor.Actor: animowane modele postaci.

- direct.gui.OnscreenText.OnscreenText: wyświetlanie tekstu w interfejsie UI.

- direct.showbase.DirectObject.DirectObject: podstawowe wsparcie dla obsługi zdarzeń klawiszy/myszy.

- direct.filter.CommonFilters: dodatkowe filtry post-processingowe (tu importowane globalnie).

- sys: dostęp do funkcji systemowych (wyjście z aplikacji).

- os: operacje na ścieżkach i plikach.

---

2. Funkcje pomocnicze

2.1 addInstructions(pos: float, msg: str) -> OnscreenText

Tworzy i wyświetla w rogu ekranu tekst instrukcji.

- Parametry:

  - pos: odległość pionowa od górnej krawędzi ekranu (w jednostkach normalizowanych).
  - msg: treść wyświetlanego komunikatu.
- Zwraca: instancję OnscreenText utworzonego napisu.

2.2 addTitle(text: str) -> OnscreenText

Tworzy i wyświetla w prawym dolnym rogu ekranu tytuł lub komunikat.

- Parametry:
  - text: treść wyświetlanego tytułu.
- Zwraca: instancję OnscreenText utworzonego napisu.

---

3. Klasa BumpMapDemo

Klasa dziedziczy po ShowBase i odpowiada za inicjalizację sceny, sterowanie kamerą, oświetlenie oraz przełączanie shaderów.

```
class BumpMapDemo(ShowBase):
    def __init__(self):
        # konfiguracja bump mappingu i uruchomienie silnika
        loadPrcFileData("", "parallax-mapping-samples 3\nparallax-mapping-scale 0.1")
        ShowBase.__init__(self)
        ...
```

3.1 __init__(self)

- Ładuje konfigurację bump mappingu (skala i liczba próbek).

- Sprawdza wsparcie sterowników dla shaderów. Jeżeli brak, wyświetla komunikat i kończy inicjalizację.
- Ustawia tytuły i instrukcje sterowania:
  - ESC: wyjście z aplikacji.
  - Ruch myszy: obrót kamery.
  - W/S/A/D: poruszanie się w czterech kierunkach.
  - Enter: przełączanie bump mappingu (wł./wył.).
- Ładuje model models/abstractroom i dodaje do sceny.
- Wyłącza domyślną kontrolę myszy/stereoskopii (disableMouse).
- Ukrywa kursor (WindowProperties().setCursorHidden(True)).
- Ustawia szeroki kąt widzenia (110 stopni).
- Inicjalizuje wektor focus, kąt heading i pitch, stan klawiszy ruchu oraz zmienną do liczenia czasu (last).
- Dodaje zadanie taskMgr.add(self.controlCamera, "camera-task") do pętli klatek.
- Powiązuje zdarzenia klawiszy z metodami kontrolnymi.
- Tworzy pivot (lightpivot) do obracania źródła światła punktowego.
- Dodaje PointLight oraz AmbientLight do sceny.
- Ładuje dodatkowy model (models/icosphere) w punkcie świetlnym, by zobaczyć jego pozycję.
- Włącza automatyczne shadery (self.room.setShaderAuto()).
- Ustawia flagę self.shaderenable = 1.

3.2 setMovement(self, direction: str, value: bool) -> None

Aktualizuje stan ruchu kamery.

- Parametry:

  - direction: klucz ze słownika {'forward','backward','left','right'}.

  - value: True lub False (naciśnięty/zwolniony klawisz).

3.3 rotateLight(self, offset: int) -> None

Obraca pivot światła o zadaną wartość.

- Parametry:

  - offset: krotność kroku (np. -1 lub 1), mnożona przez 20 stopni.

3.4 rotateCam(self, offset: int) -> None

Zmienia kąt głowienia kamery (heading) o określoną wartość.

- Parametry:

  - offset: krotność kroku (np. -1 lub 1), mnożona przez 10 stopni.

3.5 toggleShader(self) -> None

Przełącza bump mapping (shadery) i aktualizuje instrukcję na ekranie.

- Wywoływane po naciśnięciu Enter.

- Jeżeli shadery są włączone, wyłącza je (setShaderOff()), a następnie ustawia tekst przycisku na Turn bump maps On.

- W przeciwnym wypadku włącza je (setShaderAuto()) i zmienia tekst na Turn bump maps Off.

3.6 controlCamera(self, task: Task) -> Task.cont

Główna pętla sterowania kamerą, wywoływana w każdej klatce.

1. Pobiera ruch myszy z pozycji kursora.

2. Przesuwa kursor na środek okna, by obliczyć różnicę ruchu.

3. Aktualizuje kąty heading i pitch, ograniczając pitch do zakresu [-45,45] stopni.

4. Oblicza wektor ruchu w oparciu o naciśnięte klawisze:

   - forward / backward porusza wzdłuż kierunku, w którym patrzy kamera.

   - left / right porusza wzdłuż wektora bocznego.

5. Normalizuje wektor i skaluje go w zależności od czasu od ostatniej klatki.

6. Aktualizuje pozycję kamery tak, aby patrzyła na focus z pewnej odległości.

7. Ogranicza współrzędne focus do zadanego obszaru (klocki pokoju).

8. Zwraca Task.cont, by kontynuować zadanie.

---

4. Uruchomienie aplikacji

Na końcu skryptu tworzona jest instancja klasy BumpMapDemo i wywoływana metoda run():

```
demo = BumpMapDemo()
demo.run()
```

Jest to standardowa forma startu aplikacji Panda3D.

Dokumentacja przygotowana na podstawie analizy kodu źródłowego.